MSX
ビギナーズBASIC

ファースト・プログラミングのためのレクチャーノート

児玉真之 著

アスキー出版局

HOME COMPUTER

MSX ビギナーズBASIC

児玉真之 著

ASCII
アスキーブックス

MSX
ビギナーズBASIC

ファースト・プログラミングのためのレクチャーノート

児玉真之 著

アスキー出版局

はじめに

MSXというマークの付いたパーソナル・コンピュータが，多くのメーカーから発売されています．このMSXは，メーカーや機種の違いを超えた統一規格のマシンで，低価格ながら従来のパソコンに劣らない多彩な機能を備えています．

本書は,このMSXを使ってBASICを覚えようとする人のために書かれたものです．MSXは，統一規格ということから数多くの市販ソフトが利用でき，BASICを知らなくても楽しく使うことができます．けれども，せっかくMSXと向き合うからには，自分でプログラムを作り，思いどおりにMSXを使いこなしてみたいものです．BASICの知識を持つということから，自分でプログラムを作るという，もうひとつのパソコンの楽しみ方も広がっていくのです．

本書には，初めてパソコンに触れる人でも楽しみながらBASICを理解できるように，ゲームやグラフィックなどの豊富な例を載せてあります．MSXをすでに持っている方にも，これから買おうという方にも，本書は格好の入門書となることと思います．

なお，本書のCHAPTER 5 に掲載したバイオリズム・プログラム作成については，島野俊之介氏にお手伝いいただきました．記して感謝します．

1984年2月　児玉真之

C O N T

CHAPTER 1

はじめようMSX

紹介から操作まで

CHAPTER 2

BASIC基礎講座

円の描き方, 遊び方

CHAPTER 3

楽しくプログラミング

スロットマシン, グラフ, 暗号

ENTS

CHAPTER 4

グラフィック&サウンド

ゲーム・プログラミングへの招待

CHAPTER 5

これからが本格派

バイオリズム・プログラムを作る

APPENDIX

CHAPTER 1

はじめようMSX

――― 紹介から操作まで ―――

MSX BASIC 1-1 MSXを紹介しよう

ここ数年の間，パーソナル・コンピュータは大きく成長を遂げてきました．パーソナル・コンピュータの登場によって，昔は一部の大企業でしか使うことのできなかったコンピュータが，個人でも簡単に買えるようになったのです．その技術の進歩と，コストダウンの速さには目をみはるものがある，と言えるでしょう．そして今，これまでのものよりさらに親しみやすいコンピュータ，MSXが登場しました．MSXの登場によって，コンピュータは専門家の手を離れ，ビデオやラジカセのように，私たちの暮らしに溶け込もうとしているのです．

これまでにも多くの種類のパーソナル・コンピュータが発売されてきました．そのなかには，多くのユーザーの支持を得てベストセラーとなったものもあります．こういったパーソナル・コンピュータとMSXは，いったいどのように違うのでしょうか．

ひとつは，MSXが統一規格のコンピュータであるということです．これまでに発売されたパーソナル・コンピュータの種類は，数えきれないほどです．最初はほんの3，4社しかなかったメーカーの数も，パーソナル・コンピュータの市場が大きくなるにつれて，どんどん増え続けています．競争が激しくなると，メーカーは次々と新型機を開発し，より高性能な新製品が，より低価格で発売されるようになりました．そのためにユーザーの得た利益は，はかりしれないでしょう．しかし，ここで困った問題が起きてしまいました．各メーカーが自社製品の特徴を出そうとするあまりに，同じパーソナル・コンピュータであるにもかかわらず，互換性がなくなってしまったのです．

音楽を聞くときにはプレーヤーとレコードの両方が必要です．レコードであれば，どのメーカーが作ったプレーヤーでも使えるのは当たり前です．ところが，一見当たり前に思えることが，ちっとも当たり前でなかったのがパーソナル・コンピュータの世界でした．コンピュータの場合，レコードの役割を果たすのが，ソフトウェアというもので

す．このソフトウェアを買うときには，自分の使っている機械に合ったものを指定しなければなりませんでした．A社のコンピュータを持っていたら，B社用のソフトウェアを買ってきても，使うことができなかったのです．改良を加えるということと，ひとつの規格を守るということは，相反した要求なので，ある程度は仕方のないことでしょう．しかし，パーソナル・コンピュータが普及するにつれ，この問題は無視することができないほど大きくなってきました．

そこに登場したのがMSXです．MSXはひとつの統一規格で作られたコンピュータですから，違うメーカーのマシンでも互換性があります．MSX用のソフトウェアならば，どの機種でも同様に扱うことができるのです．また，モデルチェンジも統一規格を守りながら行われるので，MSXについて勉強した知識は無駄になりません．

MSXの第二の特徴は，安いということです．価格は５万円前後ですから，これまでの機械のざっと半分です．今までのパーソナル・コンピュータは，個人で買えるといっても，ポケットマネーの範囲には収まりませんでした．コンピュータに触れたことの無い人にとって，未知の対象に投資することになるのですから，かなり勇気が必要でしょう．しかしMSXならば，まさにラジカセなみの値段です．

MSXは普通のテレビにつなげられるものも多いので，余計な出費をせずにコンピュータが楽しめます．しかもこの値段ならば，万が一ゲーム専用マシンになってしまってもあきらめがつくでしょう．もちろん，最初からゲームを目的に買う人がいても，なんの不思議もありません．

これだけの低価格でありながら，MSXのパソコンとしての性能は，かなり高いものになっています．きれいな絵を16色のカラーで描くグラフィック機能，三重和音の音楽演奏機能，高精度の計算など，どれをとっても他のパーソナル・コンピュータに比べてひけをとるものではありません．

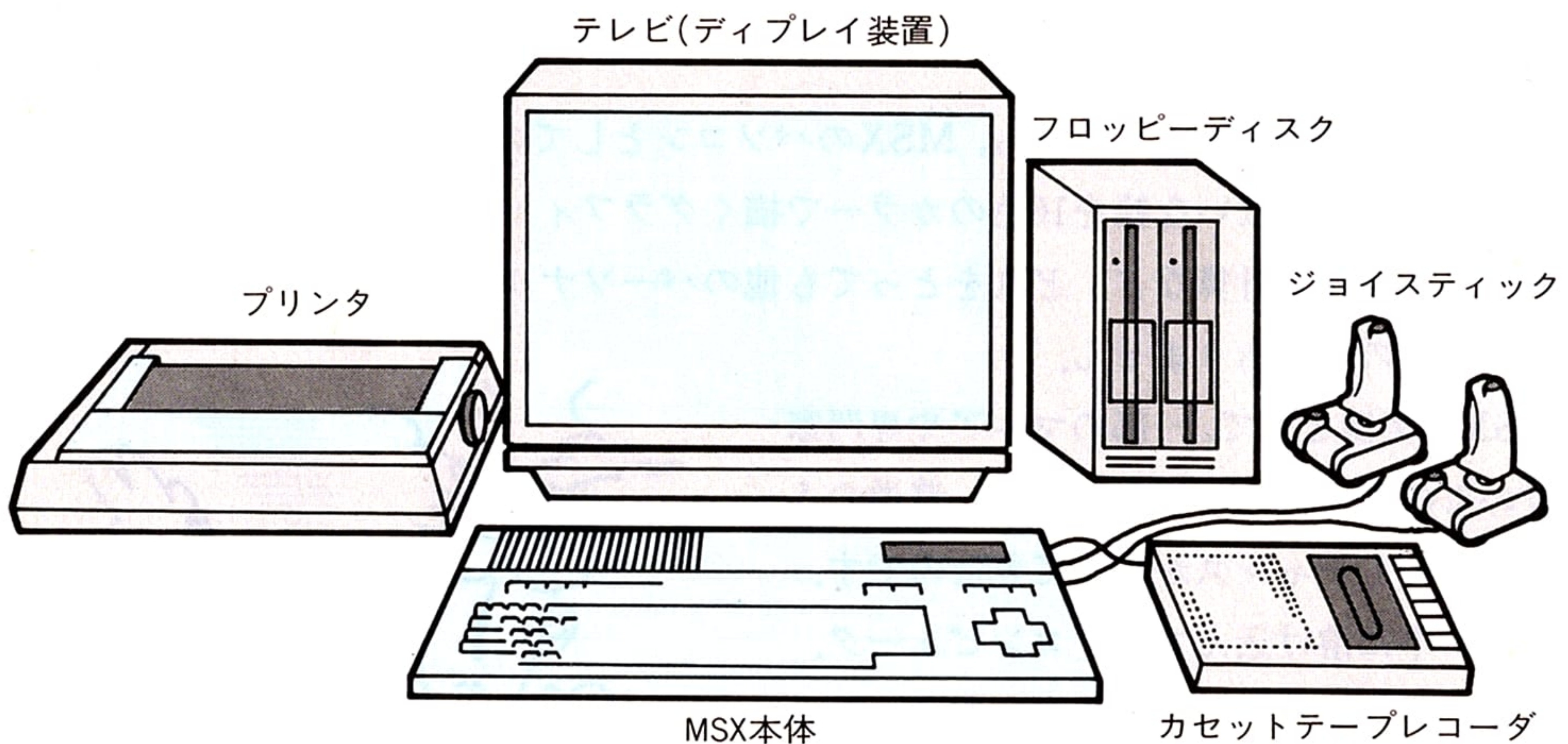
MSXを契機として，一部のマニアや専門家だけのものだったコンピュータが，普通の人の手にとどくチャンスがやってきたのです．暮らしの中に溶け込んでいくコンピュータ．そんなコンピュータとの付き合い方を，絵を描いてみたり，ゲームを作ってみたりしていくうちに，見つけ出していってください．

MSX BASIC

1-2 箱をあけてみると

MSXのすばらしさがなんとなくでもわかったら，思い切ってMSXを買ってしまうとよいでしょう．コンピュータの勉強をするとき，機械が手元にあるとないとでは大きな差があるのです．本を読むだけで得られる知識も，たくさんあります．でも，機械を触って初めて納得することもたくさんあるのです．

MSXもいろいろなメーカーから発売されています．メーカーによって多少の違いはありますが，いずれもMSXの規格を満たしているので，どれを買うかは好みの問題になります．

MSXといっても，本体の他にも必要なものがあります．次に，MSXに関連する装置をざっと挙げてみましょう．

MSXとそれに関連する装置

タイプライタに似たMSXの本体，絵や計算結果を表示するテレビ(ディスプレイ装置)や印字機(プリンタ)，データやプログラムを蓄えておくフロッピーディスク，カセットテープレコーダ，ゲームのときに便利なジョイスティックなどなど，この他にもまだまだあります．

これらのうち最低限必要なものは，本体の他にテレビとカセットテープレコーダです．とりあえず本体だけ買って，表示は家にあるテレビで代用し，データやプログラムの保存は，家のラジカセで我慢しておきましょう．

もし，家にあるテレビが，ビデオやパーソナル・コンピュータがつなげるタイプ(テレビの裏にVIDEO INと書かれた差し込みがあるもの)ならよいのですが，そうでないごく普通のテレビであれば，RFモジュレータというものが必要になることがありますので，MSXを買うお店で確かめてください．

MSXは非常にコンパクトなので，片手で持って帰ることができます．お金を払ったらすぐに帰って，さっそくMSXをセットしましょう．MSXの箱の中には，何が入っているのでしょうか．

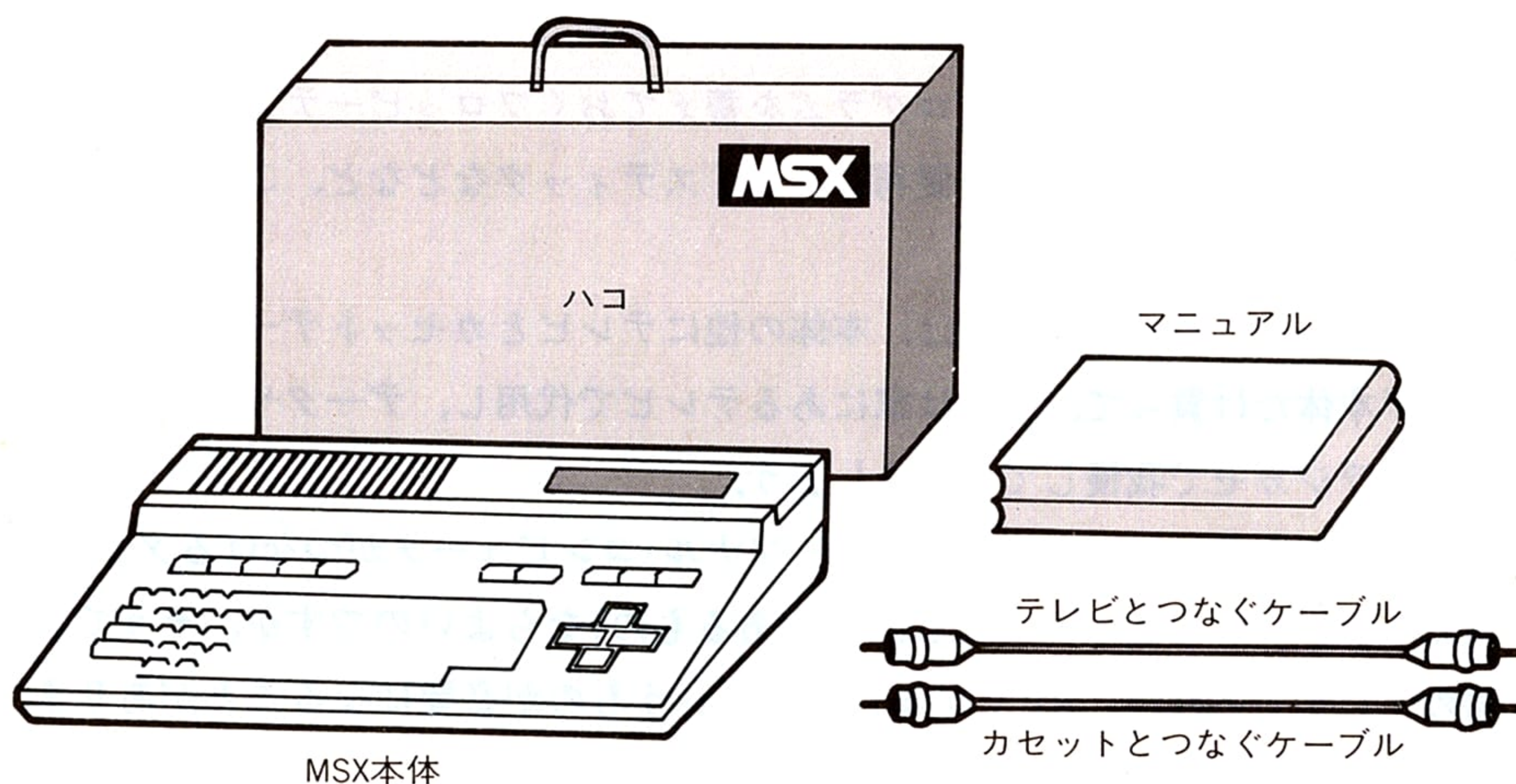
中に入っているものの詳細は，各メーカーによって違っているでしょうが，だいたい次のようなものが入っています．

ハコ

マニュアル

MSX本体

テレビとつなぐケーブル

カセットとつなぐケーブル

MSXの箱の中に入っているものの一例

○ MSX本体

タイプライタのキーのような形をしているのがMSX本体です.この中には,精密な電子部品がたくさん詰まっています.

○ テレビをつなぐケーブル

本体だけでもMSXは計算などを行いますが,その結果を表示したりするのに,テレビをつなぎます.本体とテレビをつなぐのがこのケーブルです.

○ カセットをつなぐケーブル

プログラムやデータを保存する装置として,MSXは,カセットテープレコーダを使います.カセットの録音,再生,リモートの3つのジャックにコードをつなぎます.

○ 取扱説明書

2冊ぐらいの本が入っているはずです.これにはMSXの各部分の名前から,コードのつなぎ方,それにこれから学ぶBASICというコンピュータの言葉について,こと細かに説明してあります.内容はちょっと難しいかもしれませんが,この本と併用して読めば,きっとわかると思います.取扱説明書のことをマニュアルとも呼んでいます.この本では扱わない難しい命令や,その詳細な定義を知りたいときには,マニュアルを丹念に読むとよいでしょう.

それではマニュアルを見ながら，ケーブルをつないでみましょう．これを間違えたからといってMSXが壊れるということはまずありません．安心してつないでください．次のsectionでは，電源を入れてMSXに触れてみることにします．

COLUMN

マニュアルを使いこなす

ＭＳＸを買うと付いてくるマニュアルは，たいてい２冊に分かれています．それぞれは「ＭＳＸユーザーズマニュアル」と，「ＭＳＸ　ＢＡＳＩＣ　リファレンスマニュアル」という名称で呼ばれることが多く，ページ数も200ページ前後にまとめられています．２冊のマニュアルをどんと重ねて机の上に置くと電話帳ほどの厚さになり，かなりのプレッシャーを感じるものですが，この手の本にはそれなりに読み方があるのです．そのテクニックを少し紹介しましょう．

［その１］マニュアルは通読する必要はない

マニュアルを本ではなく，辞書としてとらえましょう．困った時のさっとひと引きが，あなたの知識を増やします．

［その２］２冊のマニュアルを使い分ける

「ユーザーズマニュアル」は，家庭電化製品の取扱説明書に近いものです．ＭＳＸの接続や，ＭＳＸの操作方法などがこと細かに書かれています．それに対し，「リファレンスマニュアル」は辞書に近いもの．ＭＳＸ ＢＡＳＩＣで使える命令がアルファベット順に並んで説明されています．

［その３］「ユーザーズマニュアル」は飛ばして読む

ユーザーズマニュアルには，わかりきったことも延々と書かれていたりします．困ったことが起きてから読んでも充分間に合うでしょう．

［その４］「リファレンスマニュアル」はミクロに読む

リファレンスマニュアルは，一つひとつの命令について重箱の隅をつつくように詳しく書かれています．こちらの方は点の打ちかたひとつまで正確に読み込みましょう．

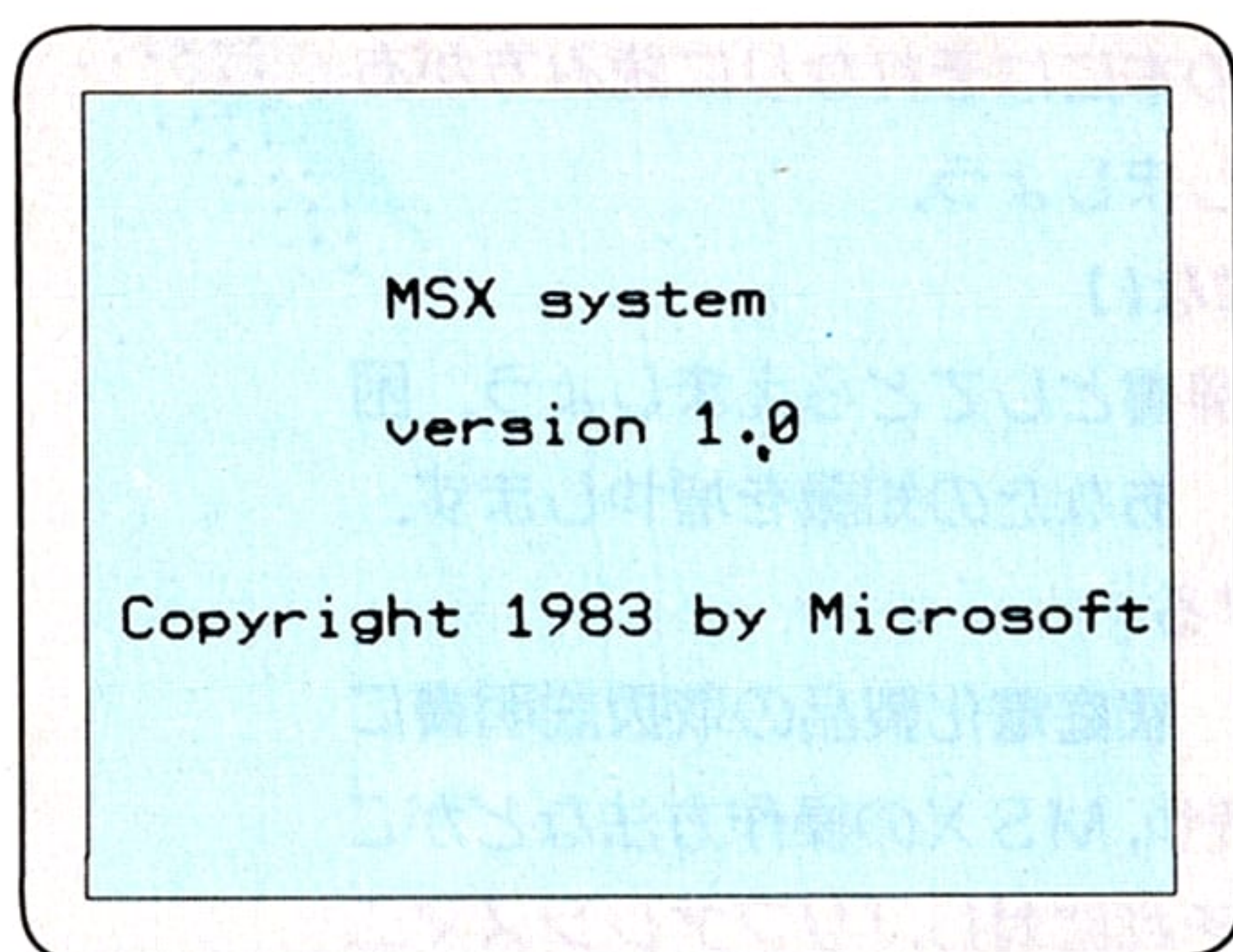
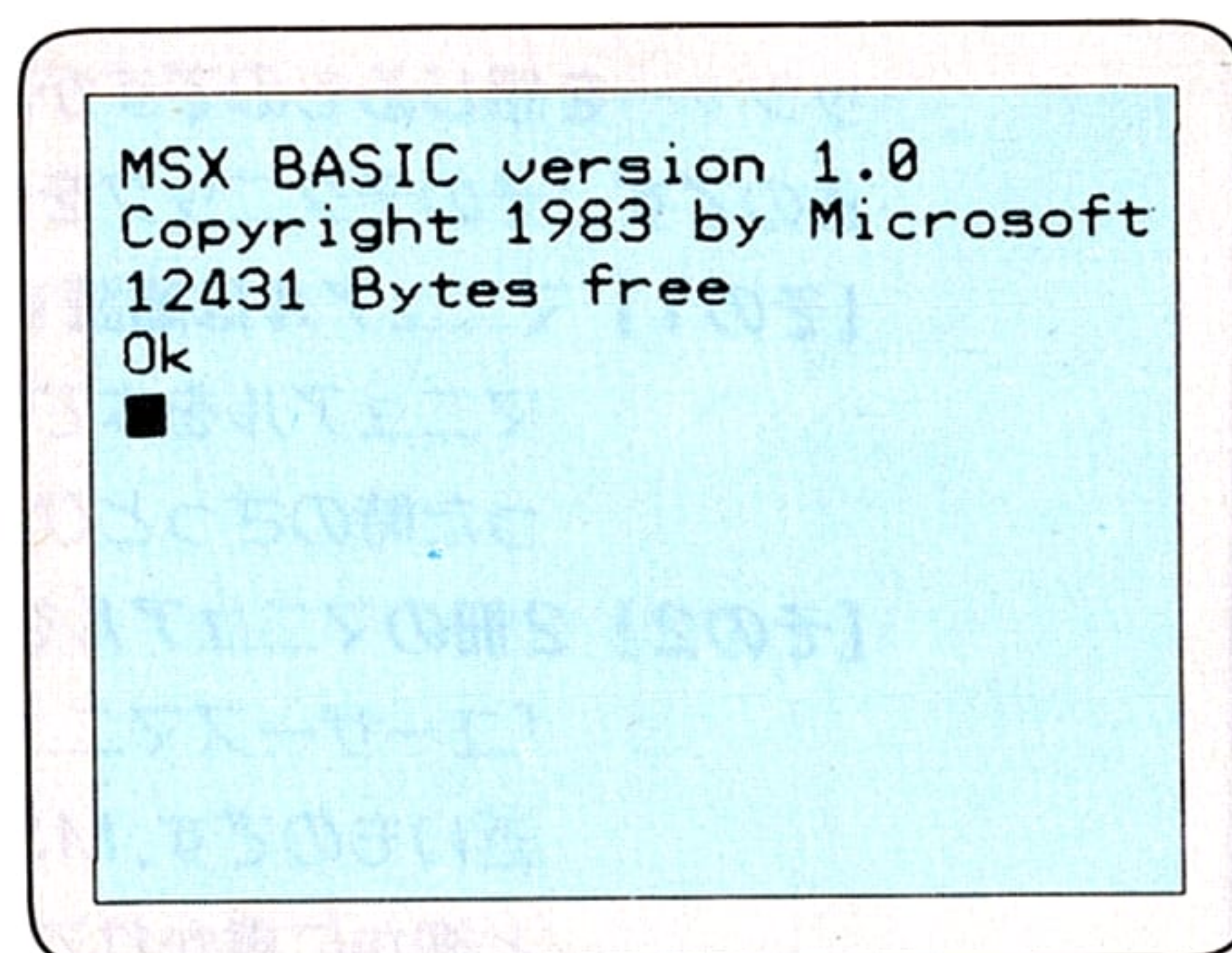
1-3 パワースイッチ・オン

ケーブルをつなぎ終ったら，準備は完了です．いよいよMSXの電源スイッチを入れてみましょう．

電源を入れる順序は，①ディスプレイ装置，②MSX本体，の順です．

```
MSX system

version 1.0

Copyright 1983 by Microsoft
```

（1）

```
MSX BASIC version 1.0
Copyright 1983 by Microsoft
12431 Bytes free
Ok
■
```

（2）

ケーブルがきちんとつながっていて，MSXが不良品でないかぎり，上のような画面がディスプレイ装置に表れてきます(一部，最初の表示が上のものと違う機種もありますが，取扱説明書を見ながら操作していくと，（2）の画面が表示されるはずです)．

ほとんどの人は，うまくいったと思います．うまくいかない人は，①コンセントが入っているか，②ディスプレイとMSX本体のケーブルの接続を間違えていないか，などをよく確かめて，もう一度試してみてください．

さて今，画面は上の（2）の状態になっています．なにやら文字が並んでいますね．これはいったい，何を表しているのでしょうか？意味のわからない英語が並んでいるので，心配になってしまう人もいるかもしれませんが，今は意味など気にすることはありませ

ん．単に電源が入って，MSXが使えるようになったしるしだという程度に考えておけばよいでしょう．

これらの表示の一番下には，

```
Ok
```

という文字が出ています．これはコンピュータが人間の指示を受け入れられる状態になっていることを表しています．「準備OK」というわけです．ここで一時画面から目を離し，MSX本体に注目してみることにしましょう．MSX本体は，下の図のようにタイプライタのような格好をしています．

STOPキー
表示を一時止めるときに押す

CLS/HOMEキー
画面をクリアしたり，カーソルを左上に戻すときに押す

INSキー
文字を挿入するときに押す

DELキー
文字を削除するときに押す

カーソル移動キー

SHIFTキー
大文字を入力するときに押す

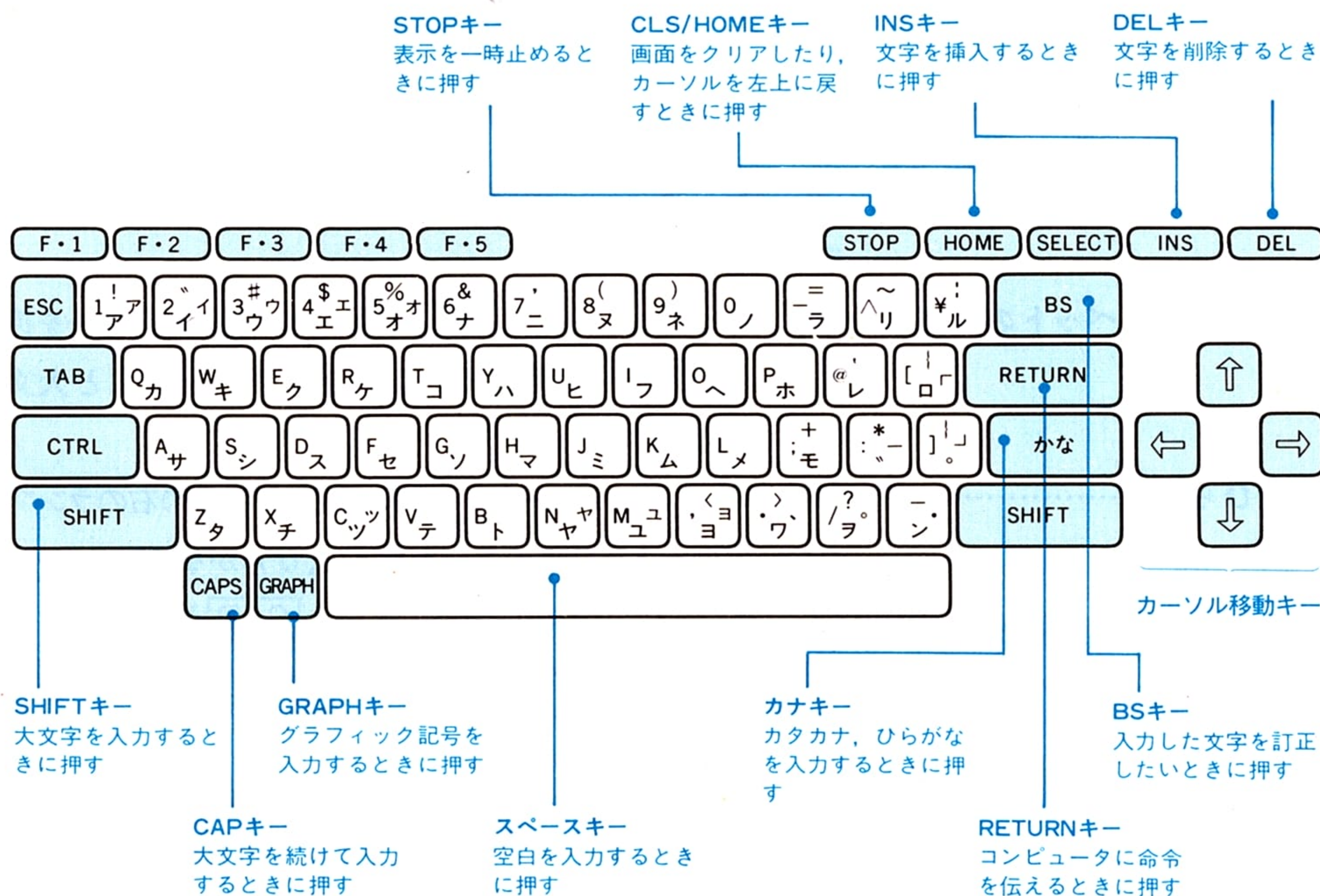
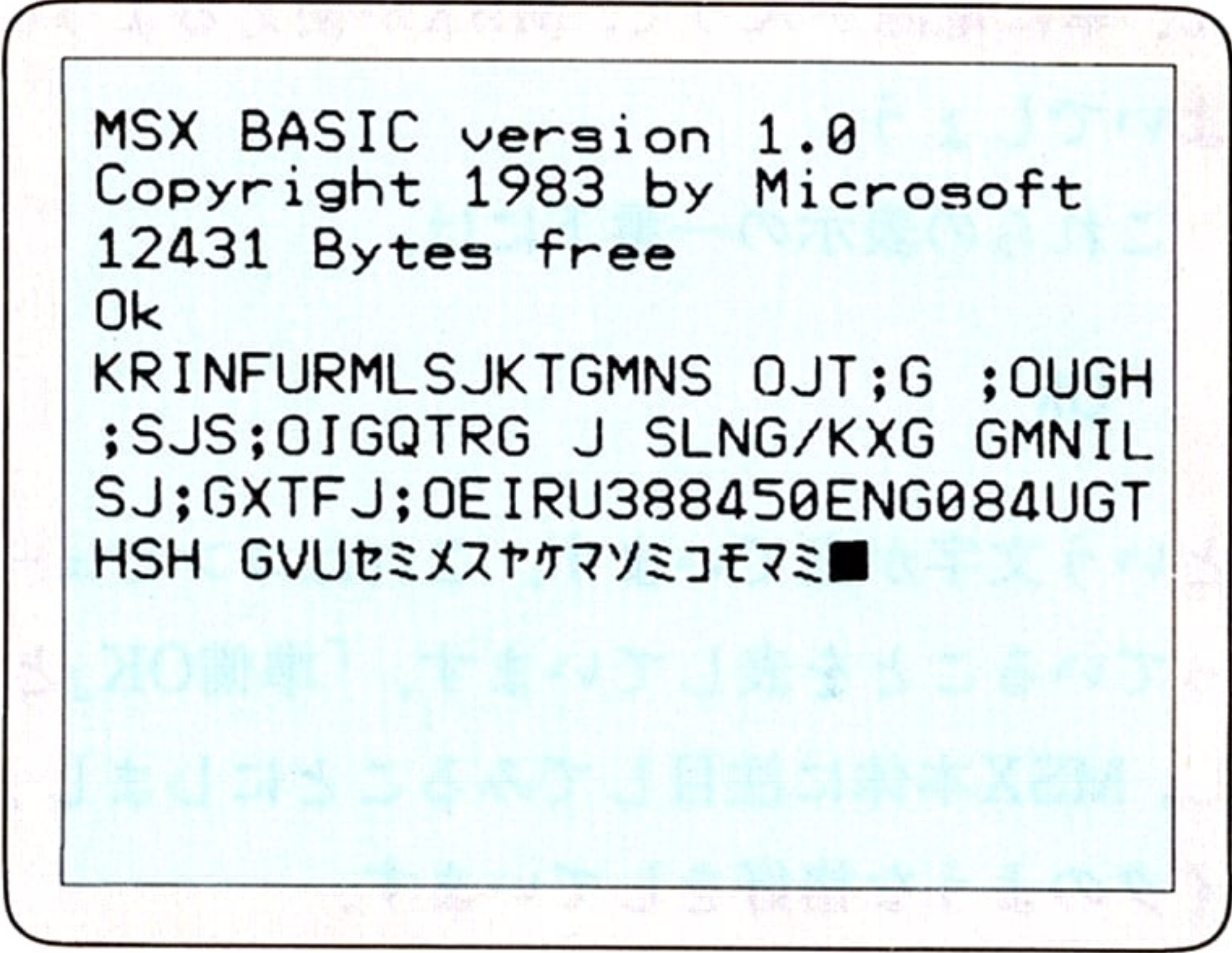
GRAPHキー
グラフィック記号を入力するときに押す

カナキー
カタカナ，ひらがなを入力するときに押す

BSキー
入力した文字を訂正したいときに押す

CAPキー
大文字を続けて入力するときに押す

スペースキー
空白を入力するときに押す

RETURNキー
コンピュータに命令を伝えるときに押す

MSXの標準的なキーボード

機種によって，デザインや配置が若干違ってきますが，おおむね，このようになっています．MSXに指示を出すには，これらのボタン(キー)を押してみればよさそうです．まずはデタラメに，キーを押してみることにしましょう．

最近のコンピュータはなかなか丈夫にできているので，ちょっとやそっと触ったくらいで，壊れることはありません．気ままにキーを押してみてください．

```
MSX BASIC version 1.0
Copyright 1983 by Microsoft
12431 Bytes free
Ok
KRINFURMLSJKTGMNS OJT;G ;OUGH
;SJS;OIGQTRG J SLNG/KXG GMNIL
SJ;GXTFJ;OEIRU388450ENG084UGT
HSH GVUｾﾐﾒｽﾔｹﾏｿﾐｺﾓﾏﾐ■
```

デタラメにキーを押した時の画面

実際に試してもらえばわかると思いますが，キーを押すと，いろいろな文字がディスプレイに表れてきます．普通の英文タイプライタであれば，アルファベットや，数字，記号しか使えませんが，MSXではもっと多くの文字を使うことができます．これをまとめてみましょう．

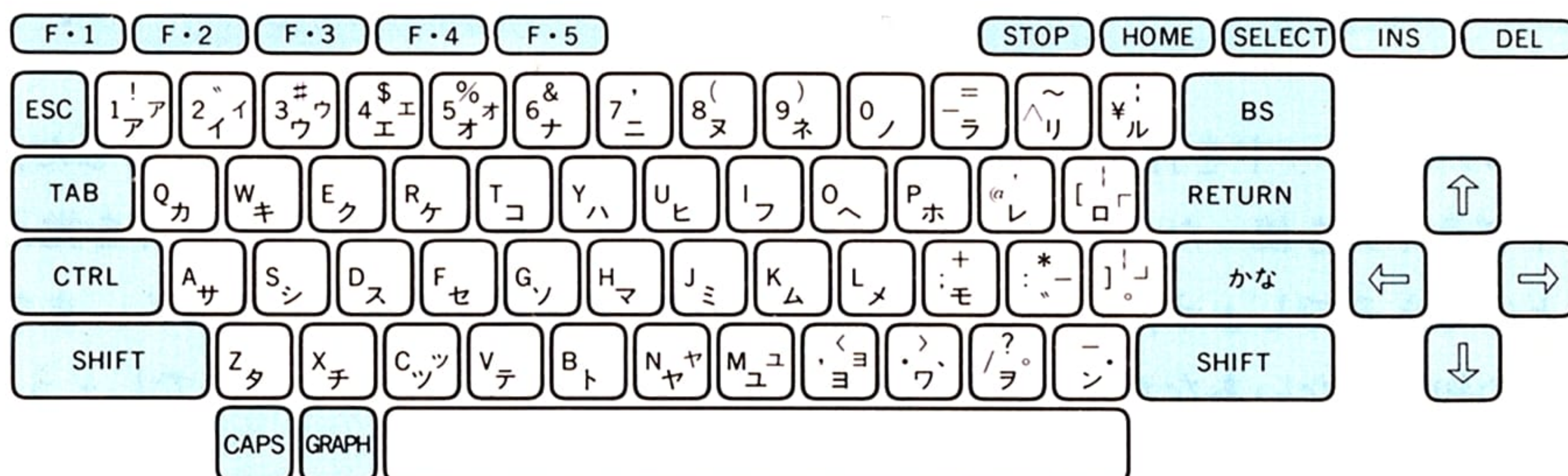
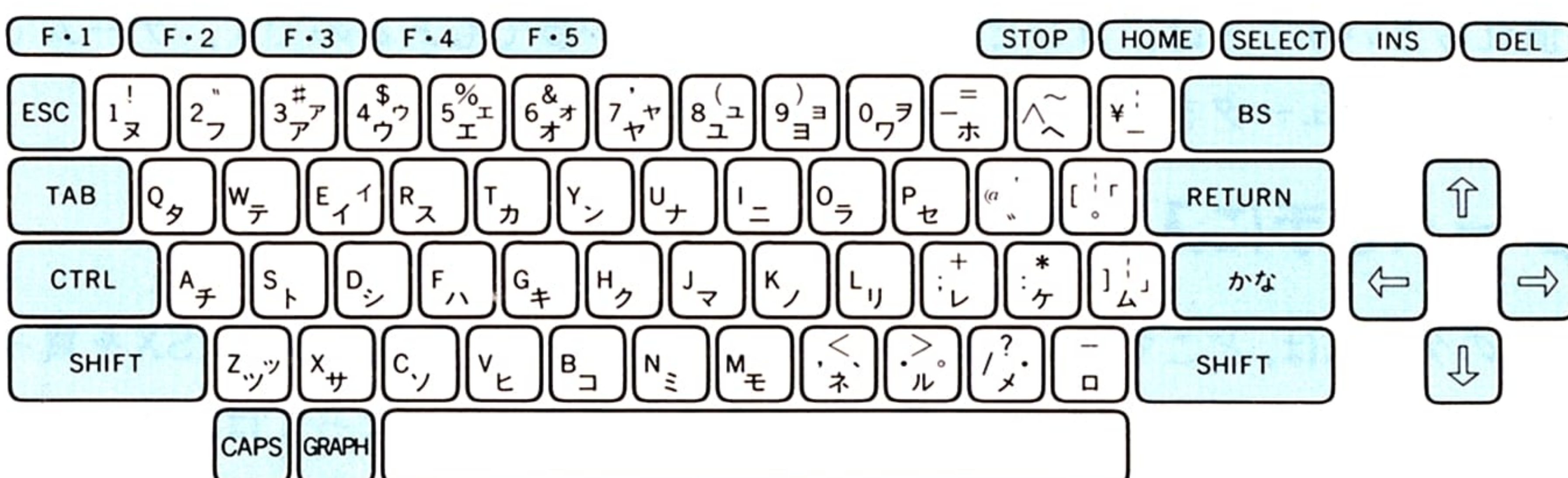
- ○**アルファベットの小文字**…(a，b，c，………キーをそのまま押すと，これが表示されます)
- ○**アルファベットの大文字**…(A，B，C，………[SHIFT]キーと一緒に他のキーを押すか，[CAP]キーが押してあるときにキーを押すと大文字になります)
- ○**ひらがな**……………………(あ，い，う，………[カナ]キーを押して，その右のランプがついているときに他のキーを押すと，ひらがなになります)
- ○**カタカナ**……………………(ア，イ，ウ，………[カナ]キーと[CAP]キーとが両方押してあると，カタカナになります)
- ○**数字**…………………………(0，1，2，………9)
- ○**記号**…………………………(&，#，；，：，………など)
- ○**グラフィック記号**…………(♣，♥，♠，◆，………など，[GRAPH]キーを押しながら，他のキーを押すと表示されます)

また，よく見るとわかるように，日，月，火……などの漢字もいくつか表示させることができます．

英文タイプライタを使ったことのある人なら，すでに気が付いているでしょうが，アルファベットの配列は，普通の英文タイプと同じになっています．英文タイプライタを

使ったことのある人は，MSXにもすぐに慣れることができるでしょう．タイプライタやコンピュータが初めて，という人も気にすることはありません．どこにどんなキーがあるかなどということは，しばらく使っているうちに自然と覚えてしまいます．

なお，ひらがな，カタカナの配列は，カナタイプと同じもの（JIS標準）と，アイウエオ順に並んでいるものがあります．

50音順キーボード図

JIS標準キーボード図

どちらの配列を採っているかは，メーカーや機種によって違っています．そして今のところ，このキーボードがあなたの意思をMSXに伝える唯一の道具です．

具体的にどうやってコンピュータに意思を伝えるのか，とか，キーボードのもっと詳しい操作の説明といったことは，この本を読み進むうちに理解できることと思います．どうしても今すぐ知りたいという人は，マニュアルの「キーボードの操作」といった項目を読むとよいでしょう．

なお，ごくまれにカチッと音がして，どのキーを押しても何も反応しなくなることがあります．このようなときは，CTRLキーを押しながら，STOPキーを押してみてください．それでもどうしようもないときは，電源を切って，もう一度入れ直してください．

MSX BASIC 1-4 ゲームソフトで遊ぼう

MSXのキーボードを押せば，画面にいろいろな文字が現れるのはわかりました．今までタイプライタを使ったことがない人でも，カタカナや♥などの表示の仕方を覚えて遊ぶことはできるでしょう．しかし，文字が出るだけで満足する人はいません．まだプログラムの知識のないあなたがMSXを能動的に使うにはどうしたらよいのでしょう．その一案は，ゲームをしてみることです．

せっかく勉強しようとしたのにゲームとは……と思う人もいるかもしれませんが，機械に慣れるということは，コンピュータの勉強の第一歩でもあるのです．ゲームで遊びながら，コンピュータを身近な存在にしていきましょう．

ゲームを手に入れる

MSXのゲームは，どこで手に入れればよいでしょうか．一番楽なのはMSXを買ったお店で，機械と一緒に何本かゲームを買ってくることです．店によっては，オマケに付けてくれるところもあるでしょう．

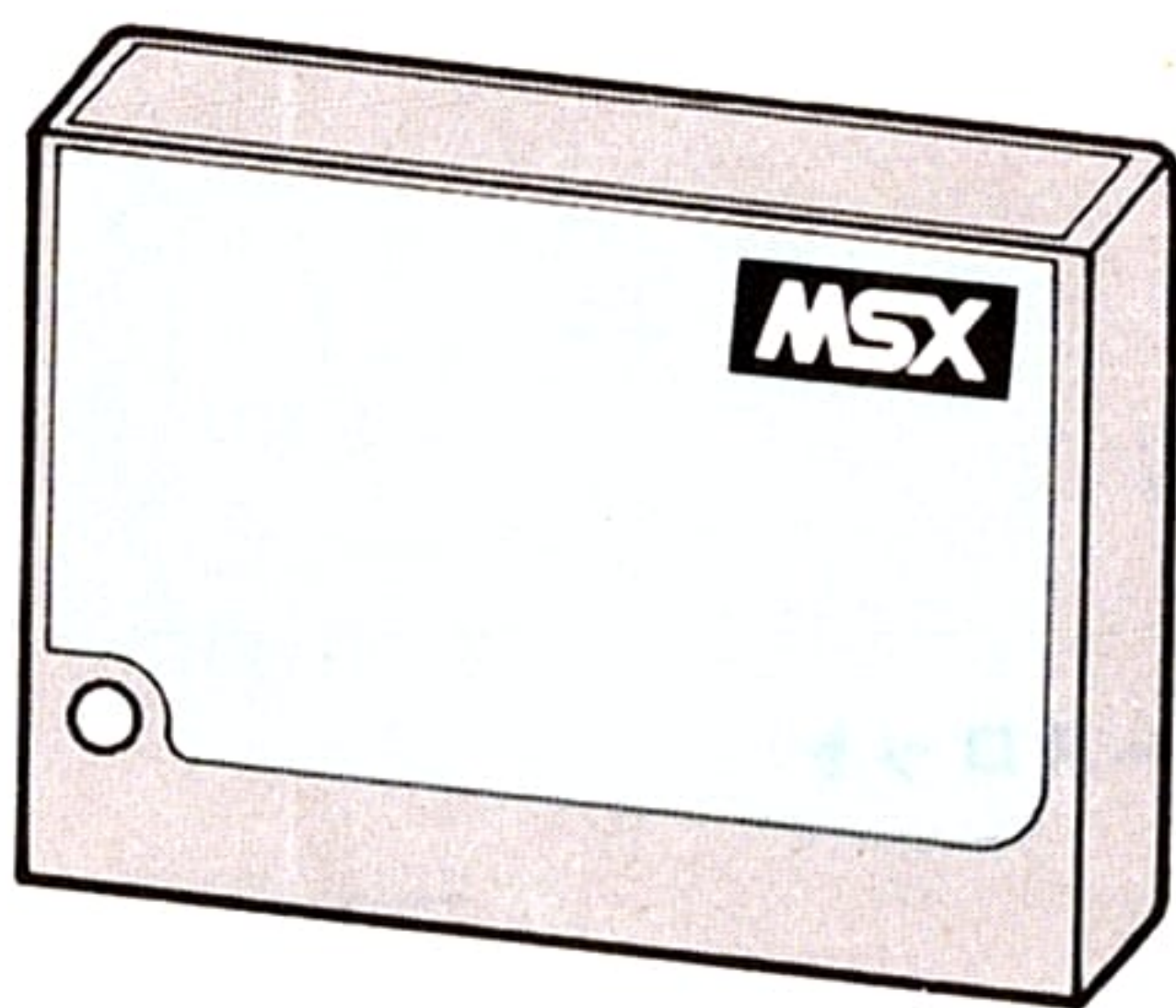
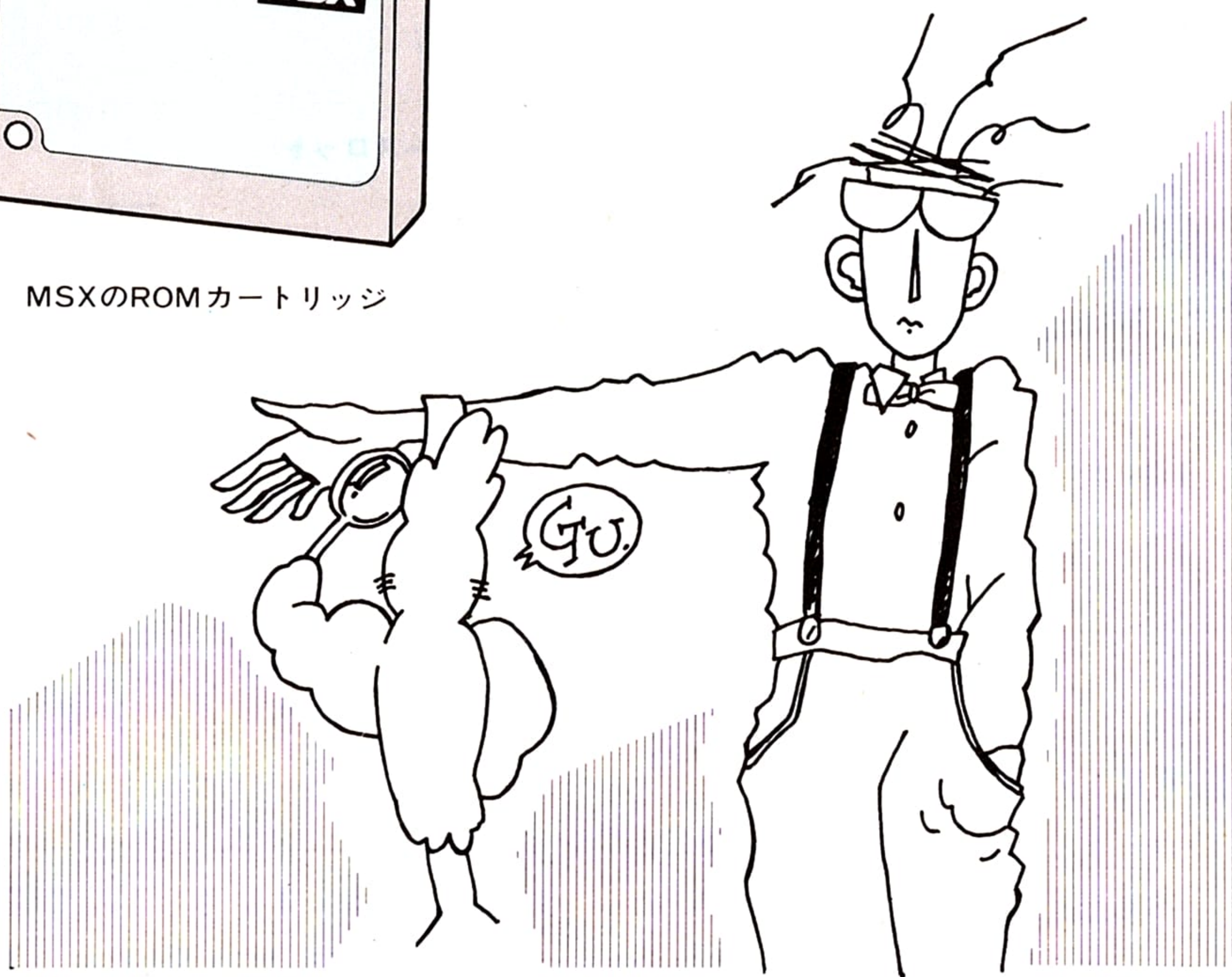
ゲームが手に入る場所としては，この他にも電気屋さん，マイコンショップ，レコード店などがありますが，最近では少し大きめの本屋さんに行くと，ゲームに限らずいろいろなプログラムがそろっています．プログラムというのは，コンピュータに仕事をさせる手順を，こと細かに整理したものです．MSXを含めてコンピュータは，このプログラムで指示されたとおりに動きます．プログラムを取り替えることによって，コンピュータはゲームマシンにもなり，家計簿にもなるのです．

さて，本屋さんなり電気屋さんなりに行ってゲームを買おうとすると，たくさんのパーソナル・コンピュータ用ゲームが並んでいます．ここでゲームのタイトルだけを見て，「あっ，おもしろそう！」と買ってしまうと，後悔することになります．これまでコンピ

ュータの規格がメーカーによってバラバラだったことは，1-1で書いたとおりです．あわててMSX用でないゲームを買ってしまったら，それはまったく使えません．MSX用のゲームには，必ず MSX というロゴマークが付いているので，このマークを確認してから買うようにしましょう． MSX マークさえ付いていれば，どのメーカーのMSXマシンにも使うことができます．

実際に見てみると，MSX用のゲームもかなりたくさんの種類があり，パッケージもさまざまです．

ROMカートリッジのゲーム

MSXのゲームは，主にROMカートリッジ，もしくはROMパックなどと呼ばれるものに入って売られています．

MSXのROMカートリッジ

この図を見ると一見何も入っていないただの箱のように見えます．しかし，実はこの中には複雑なゲームの手順を細かく記録した，ROMというLSIが入っています．このROMの中に，ゲームをするためのプログラムが書き込まれているのです．

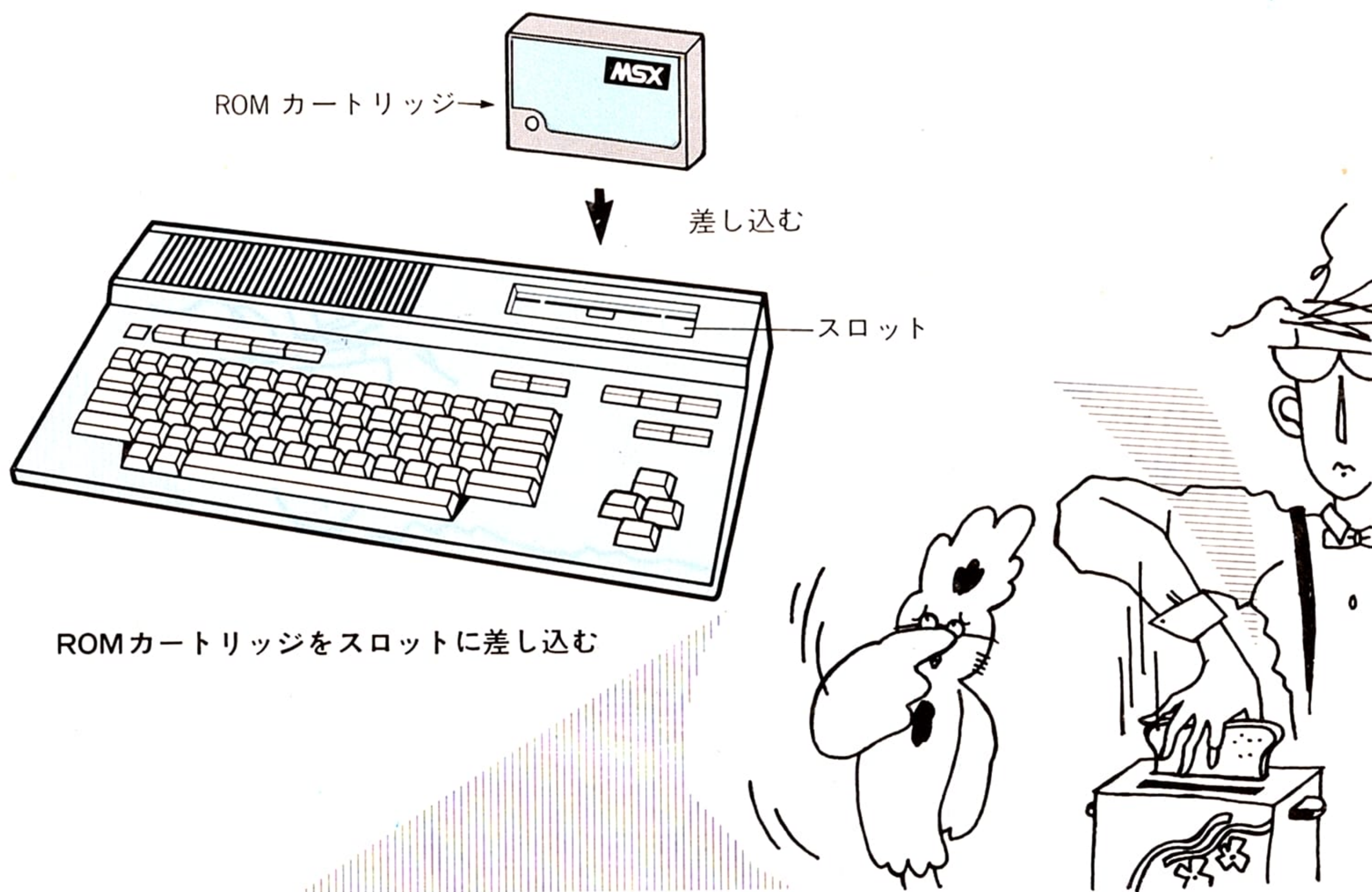
前口上はともかく，さっそくこれを使って遊んでみましょう．ROMカートリッジを使ってゲームをする手順は簡単です．

手順

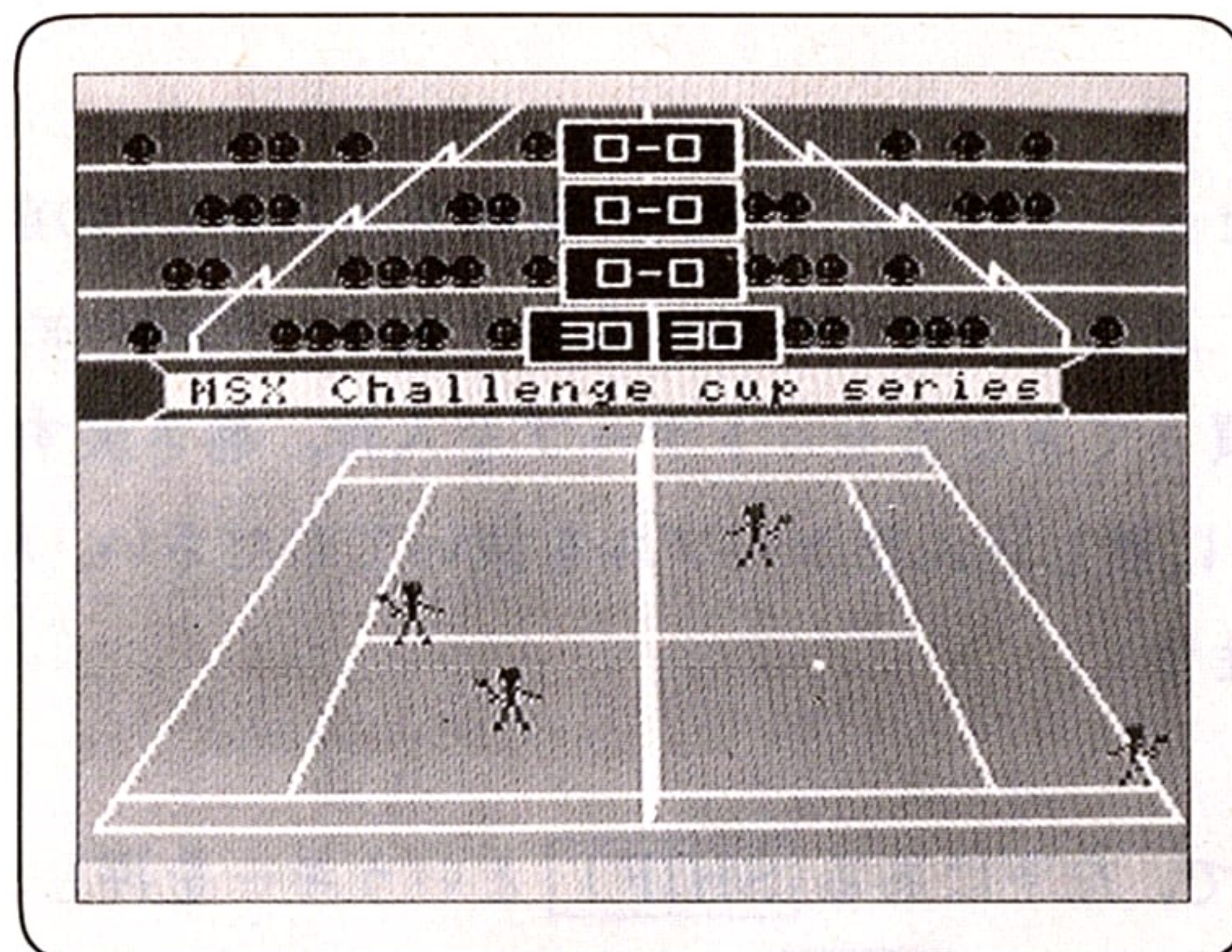
① 電源を切る．

② カートリッジを，MSX本体にある差し込み(スロットと呼んでいます)に差し込む．奥までしっかり入れてください．入れる向きを間違えると入らないようになっています．

ROM カートリッジ→
差し込む
スロット

ROMカートリッジをスロットに差し込む

③ 電源を入れる．

カートリッジを入れたら，MSXとテレビの電源を入れます．

作業はこれだけです．はやい話，カートリッジをスロットにさしこんで電源を入れるだけです．電源を入れ，しばらくすると画面が変わり，ゲームが始まります．

ROMカートリッジゲームの実行例
(「3Dテニス」アスキー)

これでMSXは，ゲームセンター並のゲームマシンに変身しました．一番右にある四方向のキー(カーソル移動キー)や，一番手前の細長いキー(スペースキー)を使いながら，ゲームを進めていきます．実際にどんなゲームをやるのかは，あなたが決めてください．MSX用のROMカートリッジに入ったゲームは何十種類もあり，新しいゲームも次々に発売されています．

カセットテープのゲーム

ROMカートリッジに入ったゲームを買ってくれば，このように手軽にゲームが楽しめます．MSXは家庭の中で誰もが使えるコンピュータを目指して開発されたもので，レコードプレーヤーに対応してROMカートリッジプレーヤーと呼べるほど，簡単に誰もがプログラムを動かせるように工夫されているわけです．しかし，ROMカートリッジの値段は高いと感じる人も多いことでしょう．なにしろ今のところ，ROMカートリッジ10本の値段とMSX本体の値段はほぼ同額です．

こんなときのために，カセットテープに入って売られているゲームもあります．このカセットテープの中に入っているものも，ROMカートリッジの中に入っているのと同じプログラムです．カセットテープは，これから説明するように，ゲームをするまでの手順が少し複雑ですが，ROMカートリッジのものより安いというメリットがあります．

それではカセットテープから，ゲームのプログラムを読み込んで，MSXでゲームを楽しんでみましょう．ここで用意するものは，カセットテープレコーダ，または専用のデータレコーダです．

手順

① MSX，テレビ，カセットテープレコーダの電源を入れる(その前に，MSXとカセットテープレコーダが，正しくつながれているかをもう一度確かめてください)．
② テープレコーダに買ってきたカセットテープを入れ，巻き戻す(動かないときには，REMOTEのところに差し込んだケーブルを抜いてください．巻き戻したあとで，また差し込んでおきます)．
③ 再生ボタンを押す．
④ MSXのキーボードで，左下にある SHIFT というキーを押しながら，上に5つ並んでいるキーの左から2番目，F・2 と書いたキーを押す．すると画面には，

cload ″ ■

と表示されるはずです．
⑤ BS と書いてあるキーを1回だけポンと押す．すると，［″］が消えるはずです．

cload■

⑥ RETURN と書いてあるキーを押す．
①～⑥の作業がきちんとできれば，カチンと音がしてテープが動き出し，しばらくすると画面に，

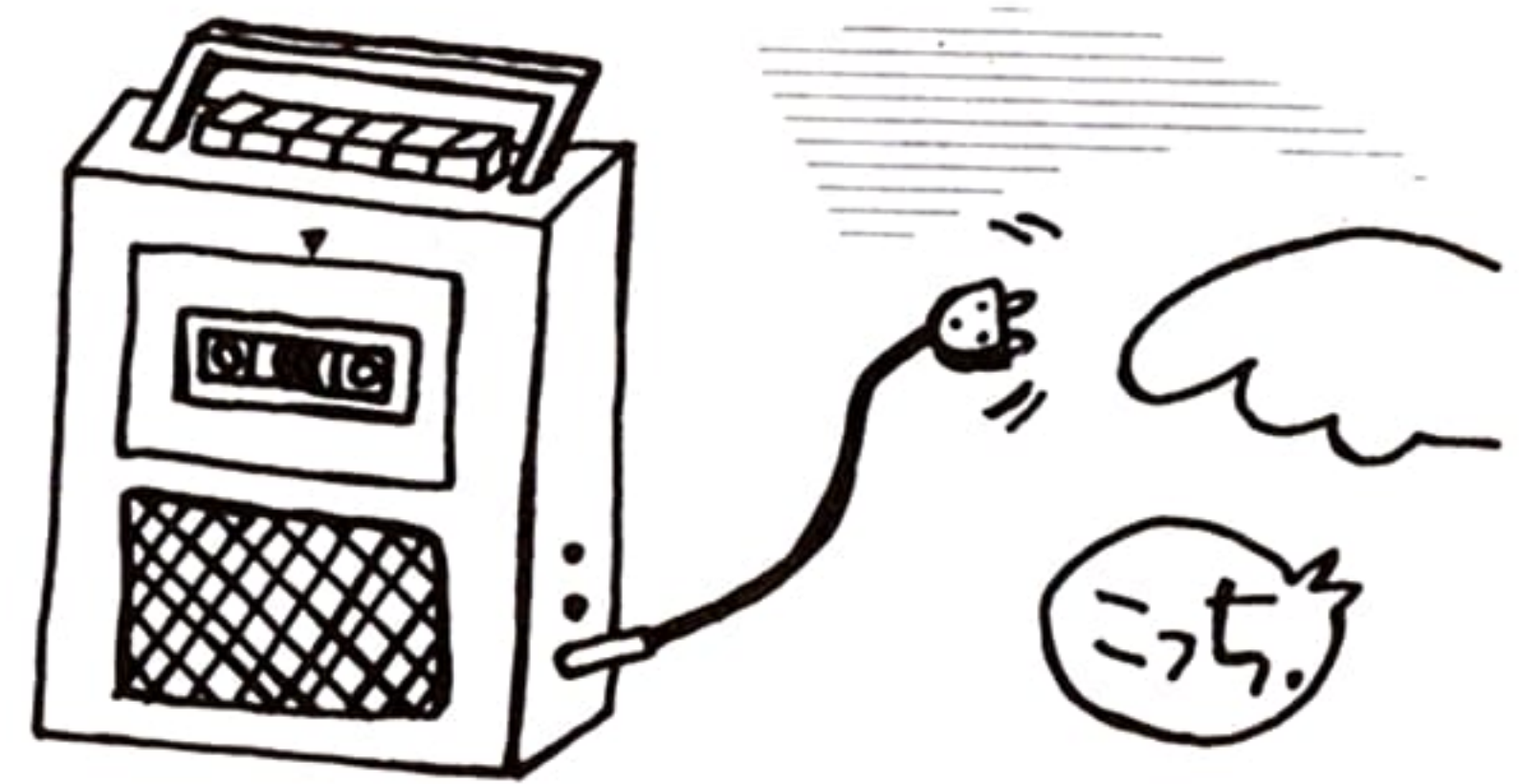
cload
Found：○○○○○

と表示されます．待つこと数分，カチッと音がしてテープが止まり，画面にOkと表示されたら，準備OKです．
⑦ 5つ並んでいるキーの一番右，F・5 のキーを押す．

これでゲームが始まります．

ところで，上のような手順に従っても，うまくいかないことがあるかもしれません．そのようなときは，いったん電源を切って，もう一度①～⑦の作業を繰り返してください．このとき注意すべき点は次の通りです．

○ **テープレコーダとMSXは正しくつながっているか**

取扱説明書を見て，もう一度よく確かめてください．赤と白のプラグが間違って差し込まれているかもしれません．

○ **ボリュームは適当か**

テープレコーダの音量が，大きすぎても小さすぎても，うまくいきません．音の大きさは，最大音量の70％ぐらいがよいでしょう．

○ **カセットテープレコーダは古すぎないか**

あまりにもボロボロのテープレコーダだと，うまくいかないことが多いようです．これを機会に最新型のラジカセを買うのもよいでしょう．

○ **テープに傷はないか**

傷があってはどうしようもありません．新品のプログラムテープに取り替えてもらいましょう．

このようにテープは手間がかかりますが，慣れてしまえばなんでもありません．

充分にゲームを楽しんだところで，いよいよ次のsectionから，基本的な事柄を覚えていくことにしましょう．

MSX BASIC

1-5 これがプログラムだ

これまでの説明にも，何回かプログラムという言葉が出てきました．新聞やテレビで，もうすっかりお馴染みの言葉です．コンピュータに仕事をさせようとするときには，人間に指示するときよりもずっと細かに，手順を指定してあげなければなりません．この，コンピュータに指示するために手順をまとめたものを「プログラム」と呼んでいます．とはいってもこれだけでは，プログラムがどういうものなのか，漠然としすぎていてつかみようがありません．もっと具体的に，あなたのMSXで使えるプログラムを紹介しましょう．

リスト

```
10 '** sample program **
20 X=100:Y=100
30 SCREEN 1,3
40 A=RND(-TIME)
50 FOR I=1 TO 32
60  READ A:A$=A$+CHR$(A)
70 NEXT I
80 SPRITE$(0)=A$
90 CLS
100 K=INT(RND(1)*4+1)
110 X=X+5-INT(RND(1)*10)
120 Y=Y+5-INT(RND(1)*10)
130 PUT SPRITE1,(X,Y),10,0
140 IF INKEY$="" THEN 100
150 PUT SPRITE0,(50,208)
160 END
170 '
180 DATA 1,3,7,15,14,12,12,14
190 DATA 15,31,31,63,127,127,3,7
200 DATA 128,192,224,240,112,48,48,112
210 DATA 240,248,248,252,254,254,192,224
```

これがプログラムです．ほとんどの人は，これを見てもわけがわからないでしょう．今はそれでもちっともかまいません．プログラムとは何なのかを知るためにも，とりあえずこのプログラムをコンピュータに記憶させ，プログラムをコンピュータが実行するとどうなるかを，見てみることにしましょう．

プログラムはキーボードから打ち込んで，コンピュータに覚えこませます．以下で説明するとおりにやれば大丈夫です．

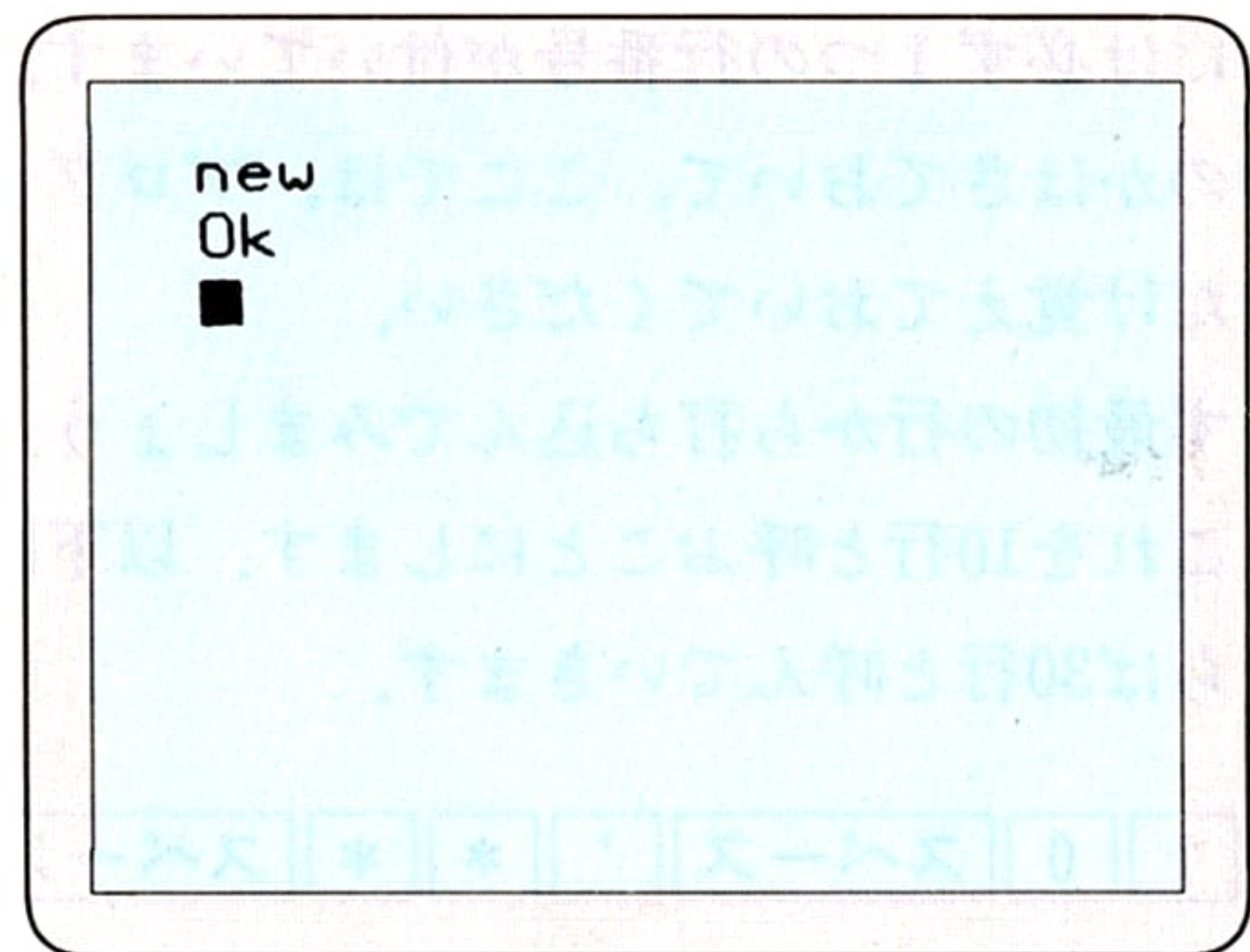
プログラムを打ち込む準備

プログラムを打み込む（入力するともいいます）前に，多少の準備が必要です．さきほど，適当にキーボードを打ちましたが，もしかすると，そのメチャクチャに打ったものが，コンピュータの中に記憶されているかもしれないのです．そこで，いったん次のように文字を打ち込んでください．

N E W RETURN

画面には右のように表示されているはずです．

```
new
Ok
■
```

このように表示されれば，コンピュータの頭の中はカラッポになって，新しいプログラムを記憶する準備ができています．

上のように表示されなかったときは，次のうちいずれかの状態にあるはずです．

○ **アルファベットの小文字でなく，カタカナが出る**

キーボードの右の方に，カナと書いてあるキーがあります．その上，もしくは横にあるランプがついているはずです．カナのキーをもう一度押すと，ランプが消えます．この状態でもう一度，N E W RETURN と打ち直してください．

○ **ひらがなが表示される**

カタカナが出るときと同じように，カナのランプがついているはずです．ランプを消してから入力し直してください．

○ アルファベットの大文字が表示される

CAP キーのランプが光っているはずです．もう一度 CAP キーを押せばCAPのランプが消え，小文字で表示されるようになります．が，大文字でプログラムを打ち込んでいっても，差し支えはありません．

○ どのキーを押しても何も表示されない

CTRL というキーを押しながら，STOP というキーを押せば，文字が入力できるようになります．どうしてもわからないときは，電源スイッチを切って，初めからやり直してください．

プログラムを打ち込む

先のプログラムを見ると，左側に数字が並んでいます．これを行番号と呼んで，１つの行には必ず１つの行番号が付いています．行番号がどういう役割で，どういう意味をもつのかはさておいて，ここでは，プログラムを行ごとに打ち込んでいくのだ，ということだけ覚えておいてください．

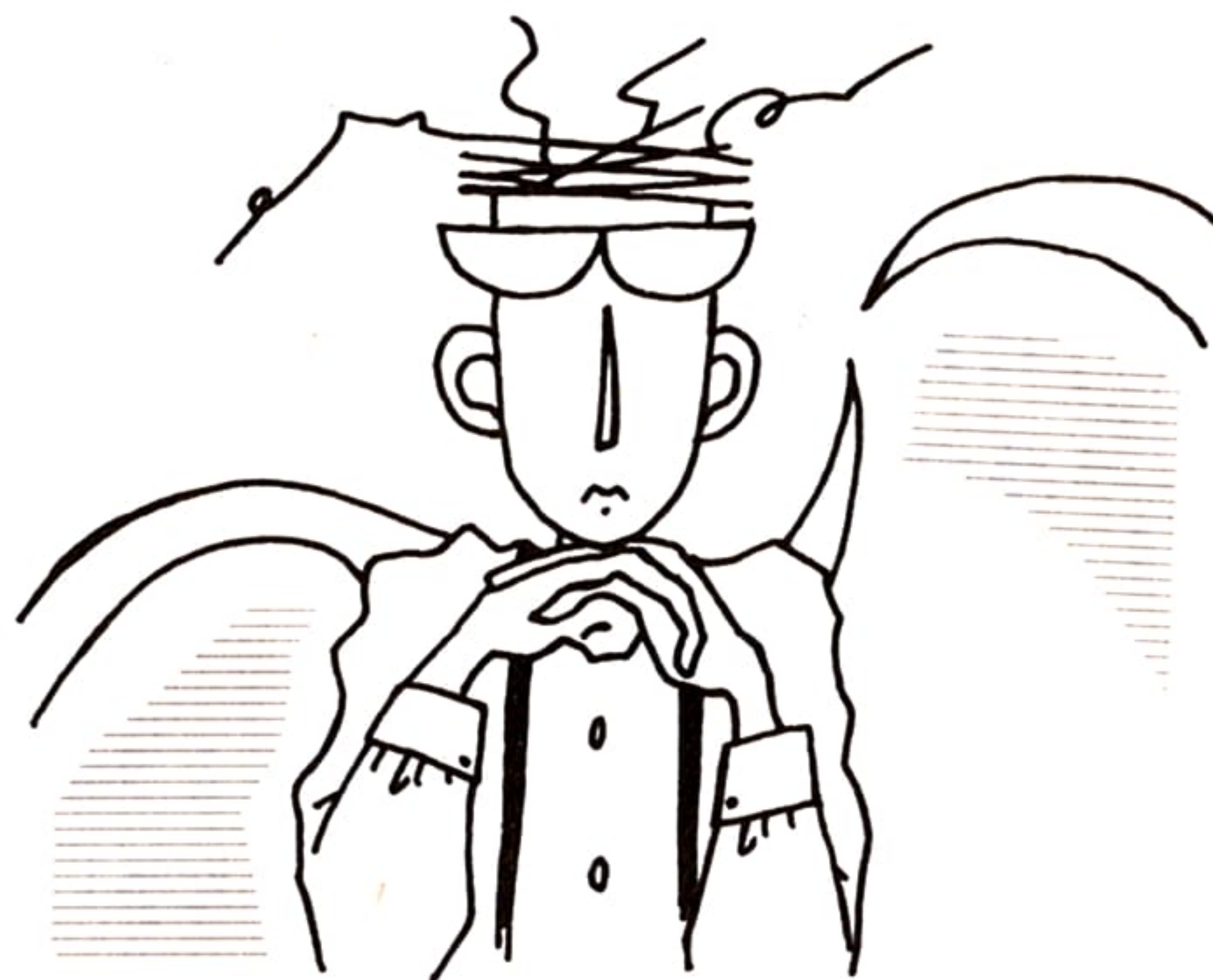
まず最初の行から打ち込んでみましょう．一番左を見ると，行番号は10になっています．これを10行と呼ぶことにします．以下同じように，行番号に20と付いていれば20行，30ならば30行と呼んでいきます．

1 0 スペース ' * * スペース S A M P L E スペース P R

O G R A M スペース * * RETURN

の順序で，実際にキーを押してみてください．

数字やアルファベットは，そのまま打ち込んでいけばよいでしょう．スペースは並びの下の方にある横長のキーを押します．

記号などでキーの表面の上に書いてあるものは，SHIFT キーを押しながら打ち込みます．' や * の打ち込み方を，次にまとめておきますので，実際にキーを押していくときに参照してください．

入力する文字	カナキーの配列	キーの押し方
,（アポストロフィ）	JIS標準	SHIFT キーを押しながら 7 ' ャ ヤ キーを押す
	50音順	SHIFT キーを押しながら 7 ' ニ キーを押す
＊（アスタリスク）	JIS標準	SHIFT キーを押しながら : ＊ ケ キーを押す
	50音順	SHIFT キーを押しながら : ＊ ー ゛ キーを押す

アポストロフィとアスタリスクの打ち込み方

もし押し間違いに気付いたら，BS キーを使います．BS キーを使って修正する様子は，下のとおりです．

BS キーを使ったプログラムの修正

10 ' ＊＊ san□　　mをnと打ち間違えた

▼　　BSを押す

10 ' ＊＊ sa□

▼　　mと打ち直す

10 ' ＊＊ sam□

▼　　続けて打ち込む

10 ' ＊＊ sample □

ひとつの行を全部打ち込み終わったら，RETURN キーを押します．この RETURN キーを忘れると，せっかく打ち込んだプログラムが，すべて無駄になってしまいます．プログラムを打ち込むときは，必ず RETURN を押すようにしてください．

同じように，20行以下も打ち込んでみてください．

2 0 スペース X ＝ 1 0 0 ： Y ＝ 1 0 0 RETURN

慣れないと時間はかかりますが，基本的な打ち込み方は，やってみればすぐ理解できると思います．

なお，どうも打ち込むのが面倒だという人は，10行と170行は省略してもかまいません．

このようにして，210行まですべてプログラムを打ち込んだら，このプログラムをコンピュータにやらせてみることにしましょう．

プログラムの実行

コンピュータを動かすには F・5 のキーを押します．このキーは，キーボードの上の方に5つ並んだうち，一番右のものです．

もし，打ち込みに間違いがなければ，下の図のようにロケットが画面の中でフワフワと踊っている絵が現れるでしょう．

しかし，よほどタイプに慣れている人ならともかく，だいたいの人は，なにかしらの間違いをしているのが普通です．

上の画面のようにならないときは，画面には，

```
Syntax error in ...
Illegal function call in ...
Subscript out of range in ...
```

などと表示されていると思います．この［・・・］の部分が，間違いのある行の番号を示

しています．間違い(エラー)の修正は，以下の手順で行ってください．

手順

① [F・4][RETURN] の順序でキーを押す．

この操作で，今打ち込んだプログラムの内容が表示されます．これをリストと呼んでいます．

```
list
10 '** sample program **
20 X=100:Y=100
30 SCREEN 1,3
40 A=RND(-TIME)
50 FOR I=1 TO 32
60  READ A:A$=A$+CHR$(A)
70 NEXT I
80 SPRITE$(0)=A$
 :
```

リストを表示したところ
(ただし，ここでは90行以後を省略している)

② 今表示されているリストは，さきほどあなたが入力したとおりになっています(ただし，小文字で打ち込んだものは，自動的に大文字になっています．アルファベットの大文字，小文字の違いはエラーと関係ありません)．これをよくよく，さきほどのリストと比べてみると，必ずどこか違っているはずです．よくある間違いは，[0(ゼロ)] と [O(オー)] の間違い，[，] と [．] の間違い，[：] と [；] の間違い，単純な打ち込みミス(タイプミス)などがあります．いずれも，ほんのわずかなことのように思われますが，コンピュータは非常に厳密で，そのような細かなミスも受け付けてくれないのです．間違いが見つかったら，間違っている行だけ，もう一度打ち込み直してください．

③ 修正が終わったら，[F・5] のキーを押して試してみましょう．うまくロケットが現れれば成功です．このロケットは，何かキーを押すと消え，プログラムの実行も終了します．

意味はよくわからないながらも，なんとなくプログラムはこういうもので，これを入れるとコンピュータが仕事をするんだということはわかったと思います．

では次から，プログラム作りの基礎を一つひとつ学んでいくことにしましょう．

COLUMN

コンピュータ・センス

コンピュータの勉強で最も重要なのは何でしょうか．それは，よくいわれているような，数学や英語の知識ではありません．むしろ，自分がコンピュータに何をやらせたいのかを明らかにし，その手順をしっかり整理できる考え方のようなものです．

自分がコンピュータにさせたいことや，その手順を整理するということは思ったほどやさしいことではありません．なぜなら人間の意識は，わかりきったことを省略して考えたり，ものごとを整理していく過程で目的を見つけたりという，論理で割り切れない構造を持っているからです．

たとえばあなたがおつかいを頼まれた時，あなたはそのために必要な手順をどの程度机上でイメージできるでしょうか．お金を持ったり，お店の場所を聞いたり，といったことはすぐにできるでしょう．しかし，服を着たり，玄関のノブを回すことまでイメージするのはかなり困難なことだと思います．

人間であれば，そんなことまで意識しなくてもおつかいをやり遂げることができます．しかし，コンピュータには人間が誰でも持っているようなこういった能力がないので，「××を買いにおつかいに行ってこい」というだけでなく「××を買ってこい．そのためにはまず服を着なければならない．服を着るためにはボタンを右手と左手を使って……」と必要なことがらすべてを細部に渡って命令しなければなりません．

その場その場で適応して目的を達成してしまう人間にとって，これらの事を意識するのはかなり無理のあることでしょう．そしてコンピュータの勉強とは，普通なら意識できないことまで意識しつくすための，センスを研ぎ澄ます訓練そのものなのです．

今書いたことは，まだコンピュータに触れてまもないあなたにとって，ピンとこないかもしれません．しかし，本書を読み進むにつれてその意味は徐々につかめてくると思います．とくにCHAPTER5では長いプログラムを作ることを例にとり，コンピュータ的なセンスの本質に触れていきたいと思います．

CHAPTER 2

BASIC基礎講座

――円の描き方，遊び方――

MSX BASIC 2-1 エンからはじまる

この章からは,実際に簡単なプログラムを,段階を追って作っていくことにしましょう.

コンピュータに指示をするものがプログラムですが,コンピュータは,人間の言葉そのものはわかりません.そこで,コンピュータの言葉を使う必要があります.コンピュータの言葉にもいろいろありますが,MSXではBASICという言葉を使います.1-5で打ち込んだプログラムも,BASICを使って書いたものです.

プログラムの打ち込み

それでは,BASICでプログラムを作っていきましょう.最初の例として,円を描くことを取り上げてみます.

```
1 screen 2
2 circle(169,95),56
3 goto 3
■
```

さっそく,上のプログラムを打ち込んでみてください.要領は,1-5でやったときと変わりありません.プログラムを打ち込む前には,

[N][E][W] [RETURN]

を忘れないように.新たにプログラムを打ち込むときは,必ずNEWとすることを習慣づけてください.

では,プログラムを打ち込んでみましょう.

1 スペース S C R E E N スペース 2 RETURN
2 スペース C I R C L E (1 6 9 , 9 5) ,
5 6 RETURN
3 スペース G O T O スペース 3 RETURN

( ) などは，SHIFT キーを押しながら，その記号のキーを押します．今回のプログラムはわずか3行ですから，打ち込みも楽ですね．

プログラムの打ち込みが終わったら，今打ち込んだBASICのプログラムを，コンピュータが確かに記憶したかを確認してみましょう．

L I S T RETURN

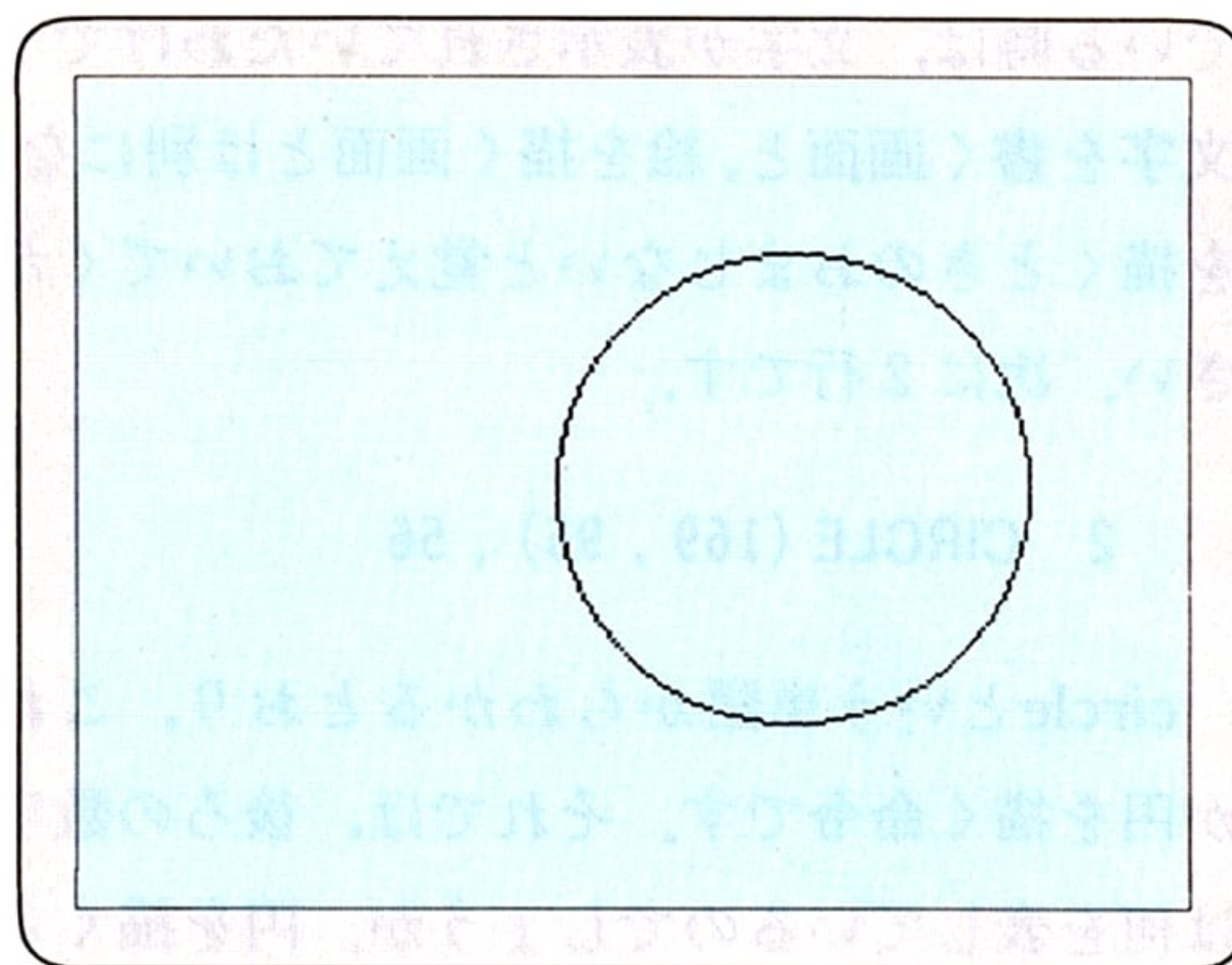
と打ち込んでください．画面上に，入力したばかりのプログラムが表示されるはずです．

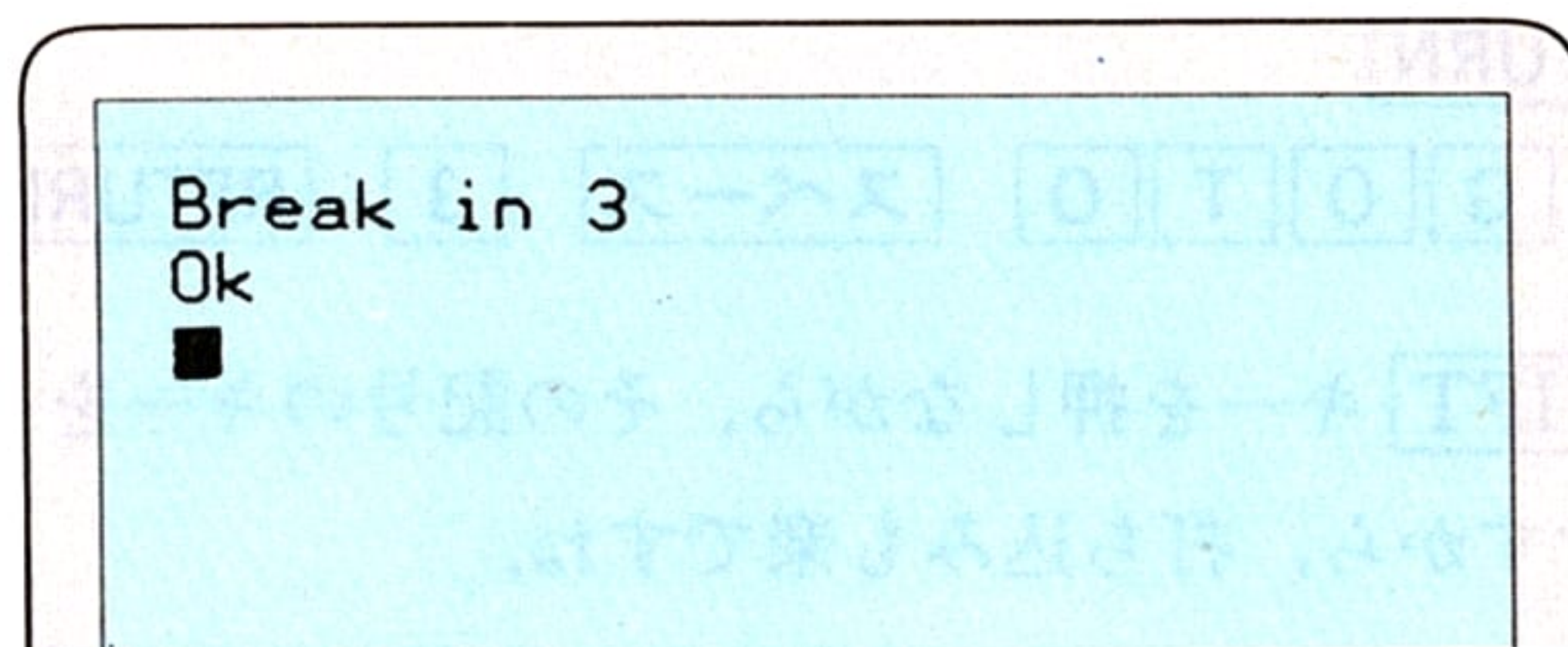
```
list
1 SCREEN 2
2 CIRCLE(169,95),56
3 GOTO 3
Ok
■
```

LISTは，コンピュータが記憶しているプログラムを，画面に表示させる命令です．小文字を打ち込んでも，リストを表示させてみると大文字に変換されてしまいます．これはコンピュータが勝手にやっていることで，あなたのミスではありません．

間違いないことを確認したら，実行してみましょう．プログラムを実行する命令はRUNです．

R U N RETURN

としてください．

あっという間に，円が画面に表示されました．プログラムを止めるときは，CTRL キーを押しながら，STOP キーを押します．円が消えて次のような画面が出ます．

```
Break in 3
Ok
■
```

プログラムの中味は？

こうしてできたプログラムの中味は，一体どのような意味をもっているのでしょうか．

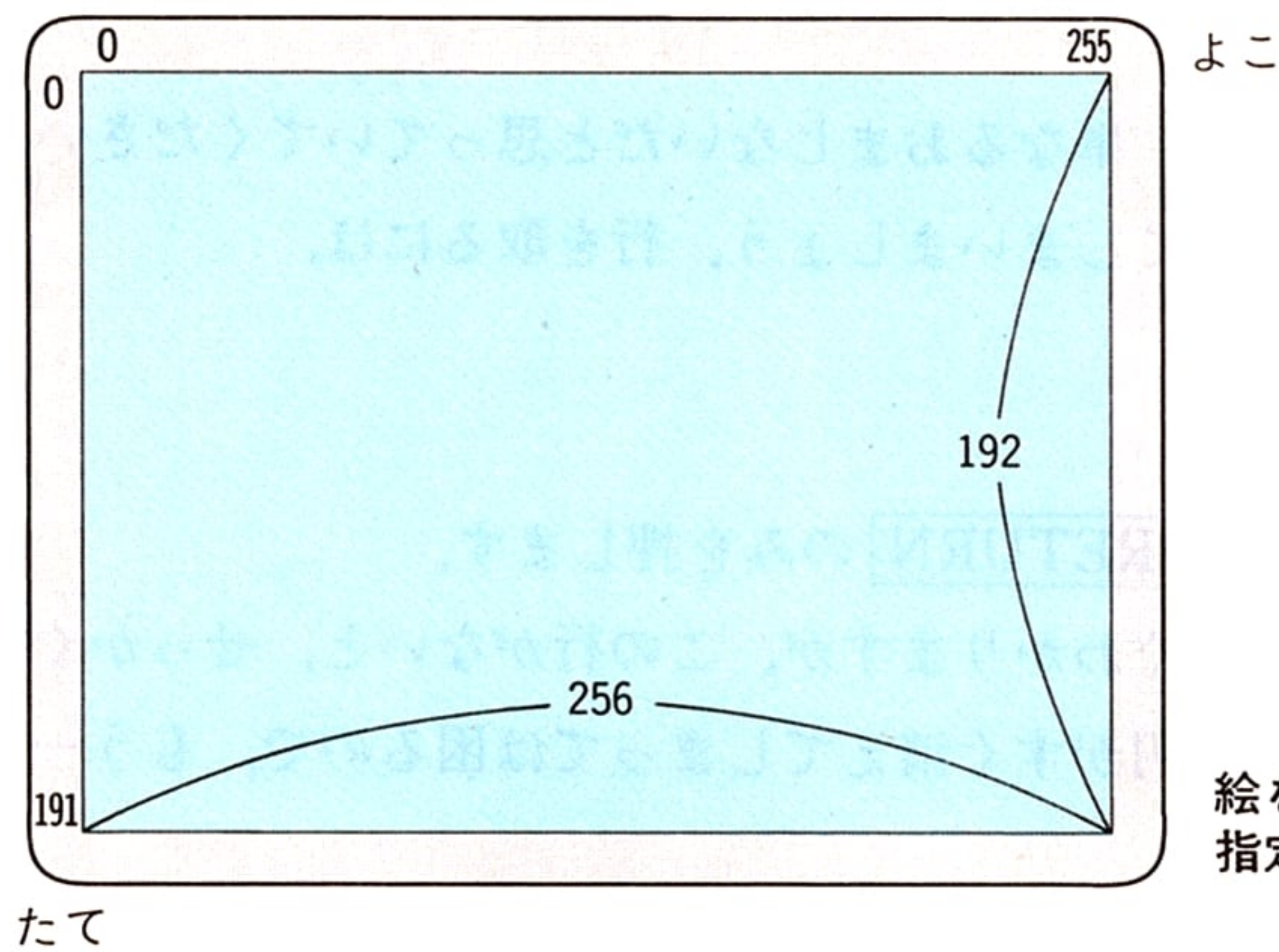
各行の左に付いている番号は，1－5で説明したとおり，行番号といいます．行番号は，コンピュータに実行の順序を指示するための番号で，番号の小さいほうから，大きいほうへと実行していきます．このプログラムの場合は，1→2→3の順序で実行されることになります．行番号は0～65529の整数を使います．－3とか0.5とかいった行番号は使えません．それでは1行から見ていきましょう．

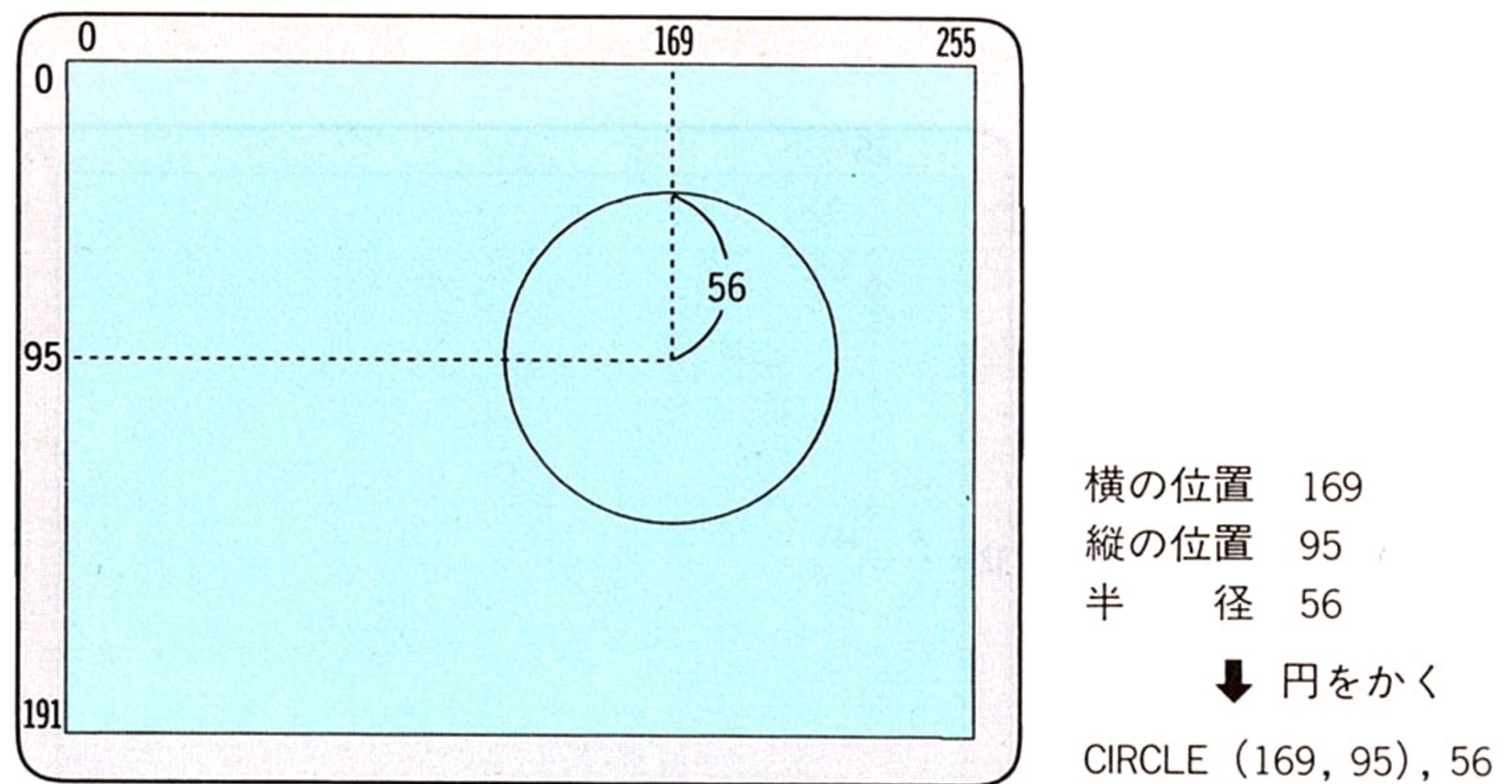
```
1 SCREEN 2
```

これは，絵を描くための画面をセットせよ，という命令です．プログラムを打ち込んでいる時は，文字が表示されていたわけですが，円は文字ではありません．MSXでは，文字を書く画面と，絵を描く画面とは別になっているのです．ここでは，SCREEN 2は絵を描くときのおまじないと覚えておいてください．次に2行です．

```
2 CIRCLE (169 , 95) , 56
```

circleという単語からわかるとおり，これが円を描く命令です．それでは，後ろの数字は何を表しているのでしょうか．円を描くときには，円を描く中心の位置と，半径の大きさを指定しなければなりませんね．

絵を描く画面で位置を指定するための座標

上の図を見てください．画面上の位置を指定するために，絵を描く画面には横 0 ～255，縦 0 ～191の番地のようなものが付いています．これを座標と呼んでいます．円を描くときにはその中心と半径を，この座標で指定します．

横の位置　169
縦の位置　95
半　　径　56
⬇ 円をかく
CIRCLE（169，95），56

今の例では，横169，縦95の位置に，半径56の円を描いてみたわけです．CIRCLE命令の一般的な型は次のようになります．

CIRCLE（横，縦），半径

さて，最後に 3 行を見てみましょう．

```
3 GOTO 3
```

この行もここでは，単なるおまじないだと思っていてください．この行がないとどうなるか，試しに取ってしまいましょう．行を取るには，

3 RETURN

のように，行番号と RETURN のみを押します．

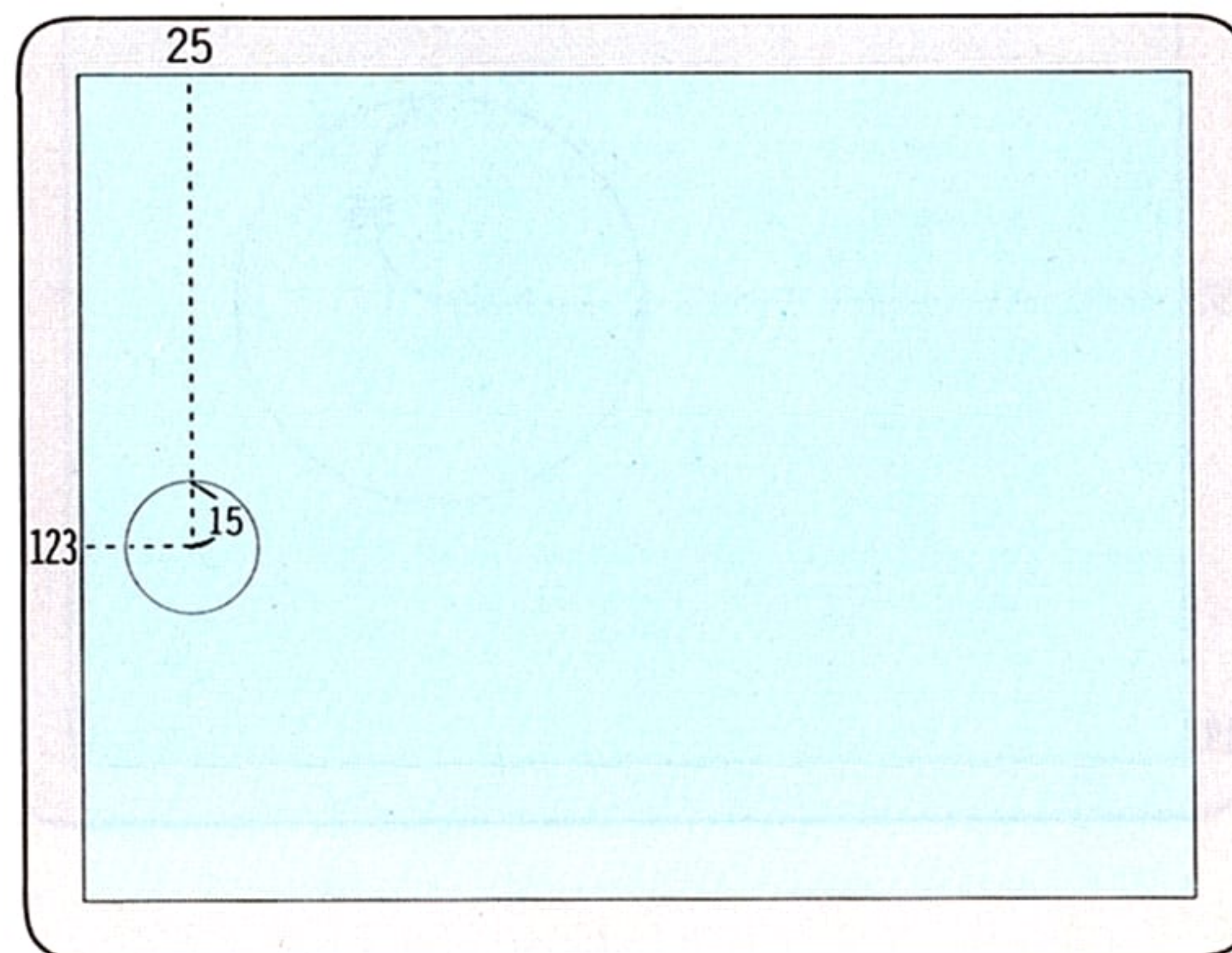
実行してみるとすぐわかりますが，この行がないと，せっかく描いた円がたちまち消えてしまうのです．円がすぐ消えてしまっては困るので，もう一度，

```
3 GOTO 3
```

と加えておいてください．

プログラムの修正

さて，別の場所に，半径の違う円を描かせるにはどうしたらよいのでしょうか．例として次のような円を考えます．

25
123
15

横の位置　25
縦の位置　123
半　　径　15

さきほどの一般型に当てはめると，上の円を描く命令は，

```
CIRCLE (25 , 123) , 15
```

となります．この円を描くために，プログラムの2行を手直しします．

ここでは，１行分を全部打ち直さずに，必要な部分だけを直す方法を覚えましょう．もう一度LISTで画面にリストを表示させてください．

リストの下にOkの文字が表れています．その下をみると■のマークがあります．これはカーソルといい，次に表示される文字の位置を示しています．カーソルは，キーボードの右の方にあるカーソル移動キーを使って動かすことができます．これを画面上で上下左右に動かして，修正したい位置を示します．修正するときには，[INS]キーや[DEL]キーも使います．これらのキーを使って修正するときの流れを，まとめておきます．

[DEL]キーと[INS]キーを使ったプログラム修正

```
2 CIRCLE(169, 95), 56
```

カーソル移動キーを使って，カーソルを〝169〞の〝1〞の上にもっていく

▼ [2][5]と打ち直す

```
2 CIRCLE(259, 95), 56
```

▼ [9]は，いらないので[DEL]キーを押す

```
2 CIRCLE(25, 95), 56
```

▼ カーソルを右に動かして９の上に持っていく

```
2 CIRCLE(25, 95), 56
```

▼ [1][2]と打つ

```
2 CIRCLE(25, 12), 56
```

▼ ひと文字分足りないので[INS]を押す

```
2 CIRCLE(25, 12), 56
```

▼ カーソルの形が変わって挿入ができる状態になったことを示している．ここで[3]を打つ

```
2 CIRCLE(25, 123), 56
```

▼ カーソルを移動して56を15に修正したら，[RETURN]を押す．カーソルは次の行の先頭に移動

```
2 CIRCLE(25, 123), 15
3 GOTO 3
```

★[DEL]キー：文字を削除するキー．カーソルのある位置の文字を削除．カーソルの位置は変化しない．

★[INS]キー：文字を挿入できるモードにするキー．挿入モードにすると，打ち込んだ文字がカーソルのある位置に挿入されていく．挿入モードは，[RETURN]を押すか，カーソル移動キーを押せば解除できる．

2-2 MSXが質問する〃

MSX BASIC

行番号を付け直す

2－1では円を描くプログラムを作ってみました．これで，自分の好きな場所に，好きな大きさの円を表示させることができるようになりましたね．

でもこのプログラムでは，たかだか円を1つ描くために，いちいちカーソルを移動させてプログラムを修正してやらなければなりませんでした．そこで，なんとか他の方法がないものか探ってみることにしましょう．

現在のプログラムは，行番号が1，2，3となっていて，行の間に命令を付け加える余裕はありません．そこで行番号を付け直しましょう．行番号を付け直す命令は，

RENUM

です．キーボードから［R］［E］［N］［U］［M］ ［RETURN］と打ち込むと，行番号が10から10おきに付け直されます．

```
renum
Ok
list
10 SCREEN 2
20 CIRCLE(25,123),15
30 GOTO 30
Ok
■
```

リストを取ってみると，確かに番号が付け直されているのがわかります．これで，行の追加がしやすくなりました．たとえば，10行と20行の間に何か入れたいときは，15行を作ればよいのです(11～19 行の 9 行の追加が可能です)．

画面上の表示を消したい時は，SHIFT キーを押しながら HOME キーを押します．画面に表示された文字を消してみてください．なおここで消えたのは，画面の上の文字だけで，プログラムそのものは消えていません．心配な人はリストを取って確認してみてください．

変数を使う

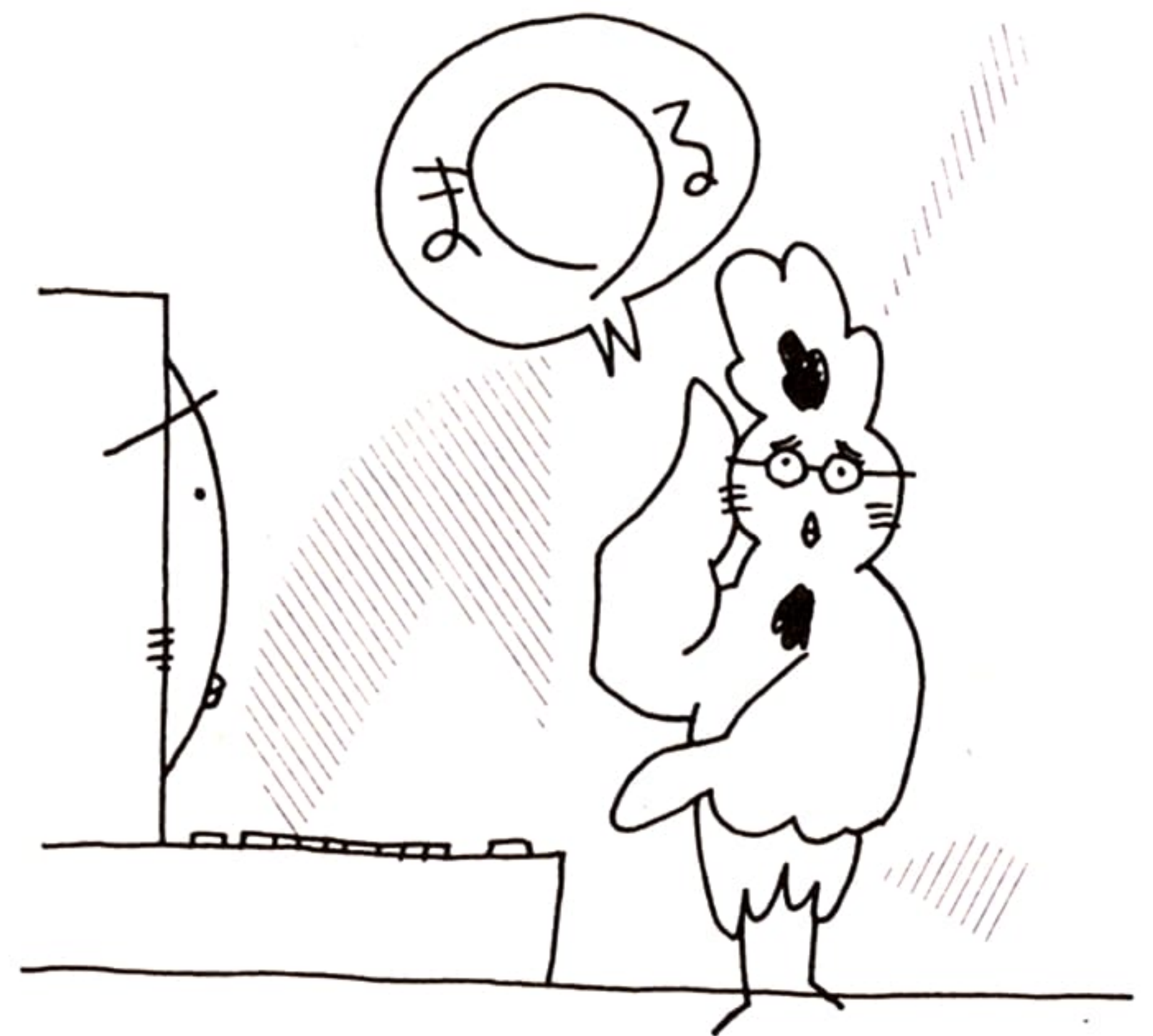
さて，次のように 3 つの行を追加してみます．

```
3  X=25
4  Y=123
5  R=15
```

追加が終わったら，20行を次のように修正してください．

```
20  CIRCLE  (X , Y) , R
```

修正は，カーソル移動キーを使って行えばよいですね．

以上の追加，修正を行ったあと，リストを取ると次のようになります．

```
list
3 X=25
4 Y=123
5 R=15
10 SCREEN 2
20 CIRCLE(X,Y),R
30 GOTO 30
Ok
■
```

きちんと内容が修正されていることと，追加した3つの行がリストに加えられていることを確認しておいてください．

以上の作業が済んだら，新しいプログラムをRUNで実行してください．実行した結果そのものは，2-1と変りありませんね．

ここで出てきたX，Y，Rなどの文字を変数といいます．変数は，数字の入った箱だと考えればよいでしょう．変数を使うことで，このように，実際に値を扱う所をひとつにまとめておくことができます．変数はなにもX，Y，Rに限るわけではなく，最初の1文字が，A～Zの英字であればかまいません．

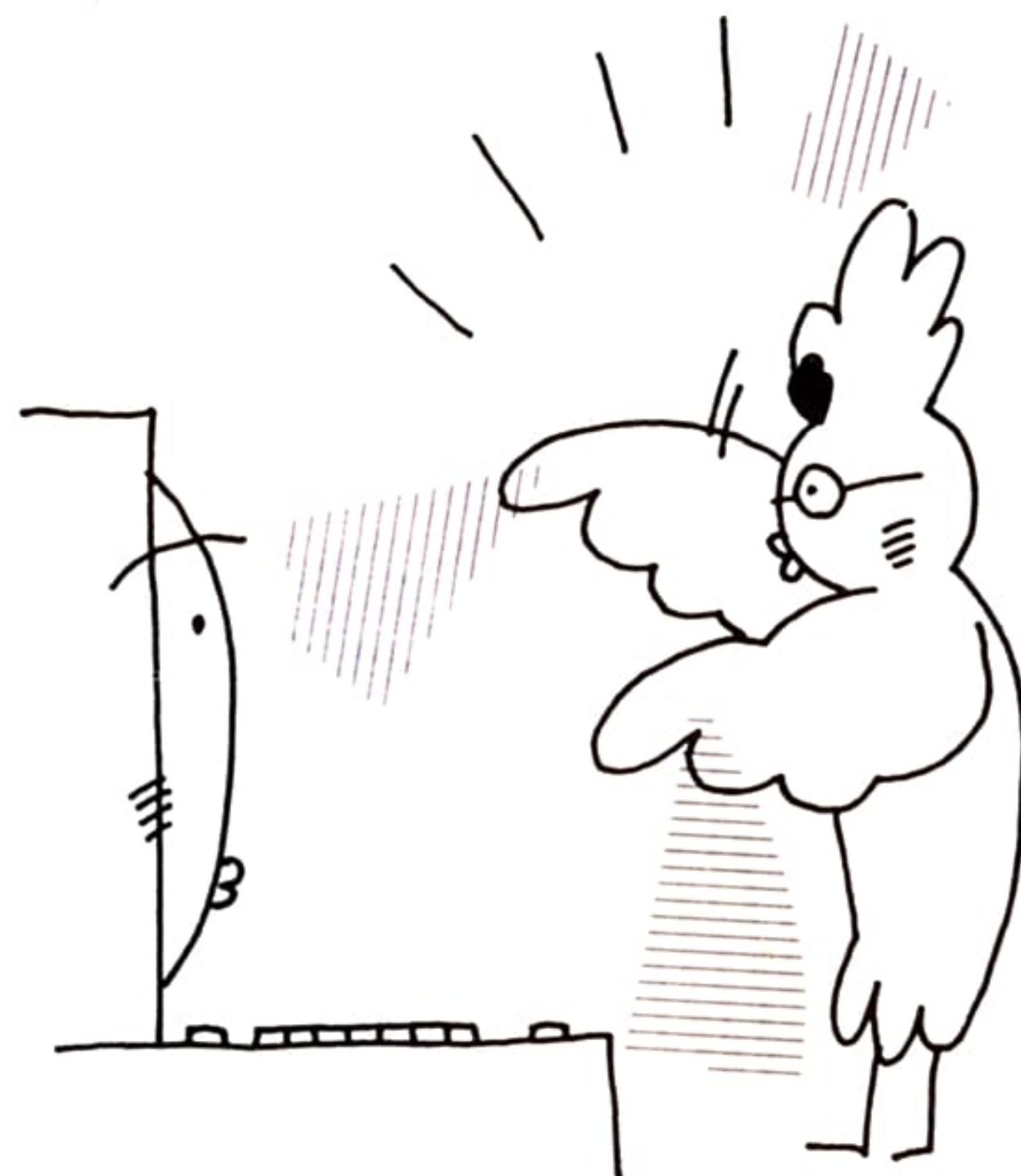
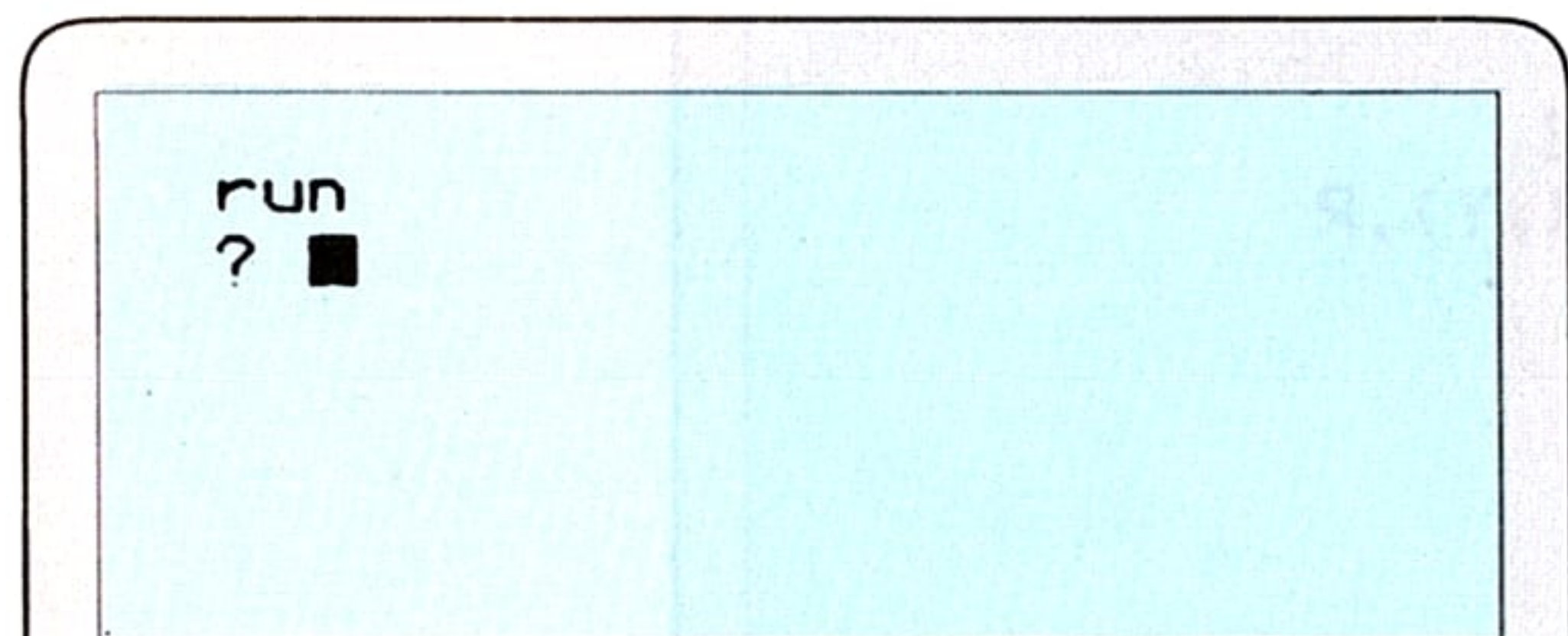
さて，変数をこのように使えば，修正の時［INS］キーや［DEL］キーをあまり使わなくてもよいというメリットはあります．実際に他の円を描かせてみればわかりますが2-1のプログラムよりは修正が簡単です．しかし，このままでは他の円を描かせたいときには，やはりプログラム自体を直さなければなりません．そこで，次に説明するように，コンピュータにX，Y，Rの値をたずねさせ，これに対して人間が値を指定できるように，工夫してみましょう．

INPUT命令

3～5行をさらに次のように修正してみます．

```
3  INPUT  X
4  INPUT  Y
5  INPUT  R
```

こうすると，果たしてどうなるのでしょうか．ちょっと想像してから実行してみてください．

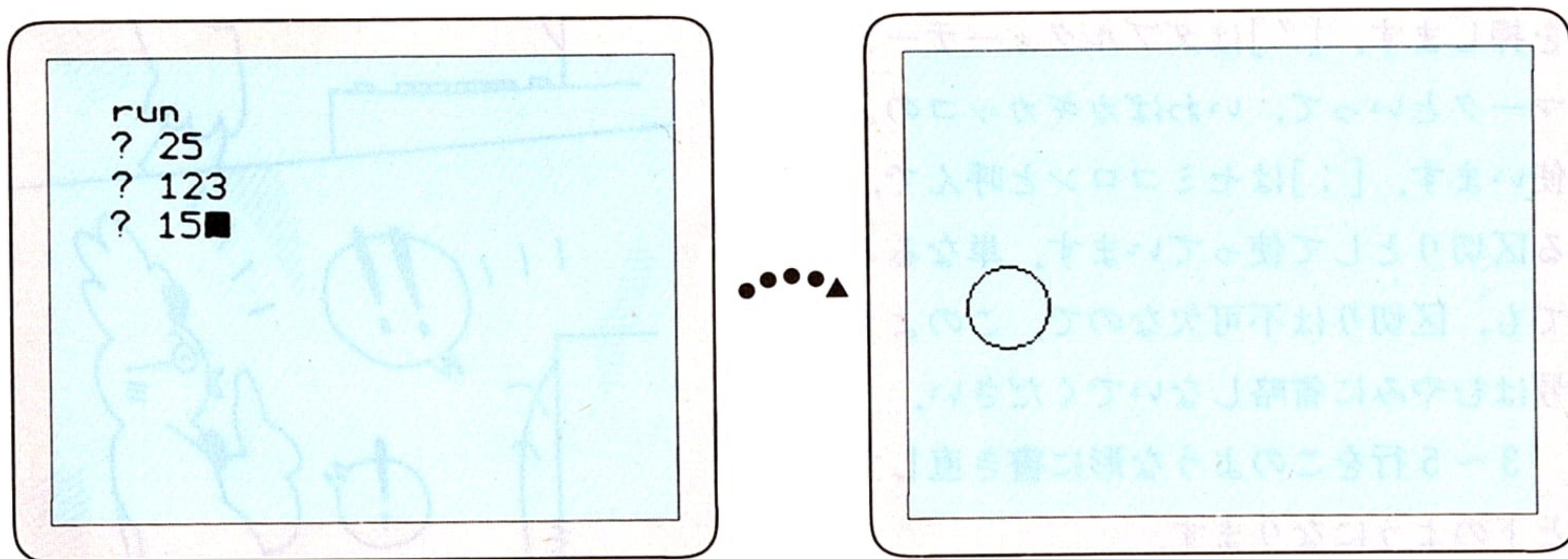
```
run
? ■
```

実行すると［?］が表れます．これはどういう意味なのでしょうか．プログラムは行

番号の小さい順に実行しています．つまりこの［？］は，3行の命令によって表示されています．3行はINPUT　Xとなっているので，X(横)の値をキーボードから打ち込めばよいようです．そこで，

[2][5] [RETURN]

としてやります．数字をコンピュータに打つときも終わりに[RETURN]を付けます．ひとつ数字を打ち込むと，再び［？］が出てきます．今度はYの値(縦)を要求しているようなので，[1][2][3][RETURN]とします．するとみたび［？］が表れます．今度はR(半径)を指定すればよいのでしょう．[1][5][RETURN]と打ち込みます．

すると，今キーボードから指定した位置に，指定した半径で円が描かれます．

```
run
? 25
? 123
? 15■
```

これまでの例でもわかるかと思いますが，キーボードから数字を入力する命令が，INPUTです．

INPUT　変数名

INPUTは上のような型で使います．これでいちいちプログラムを修正しなくても，円の位置や半径を指定することができるようになりました．いくつかの数値で，実際に試してみてください．

ところで，［？］だけでは，キーボードから何を入れたらいいのかそのうちわからなくなってしまいます．そこで，［？］だけでなく，今どの変数の値を入れればよいのかを示す文字が表示されると，ずいぶん使いやすくなります．

そのためには，INPUT命令を次の型で使います．

INPUT〝かかせたい文字〟；変数名

たとえば横の位置を指定するものであれば，

INPUT〝よこ〟；X

のようにすればよいことになります．実際にこれを打ち込むときのことを，ちょっと考えてみましょう．

［〃］は，SHIFT を押しながら 〃2 のキーを押します．［〃］はダブルクォーテーションマークといって，いわばカギカッコのように使います．［；］はセミコロンと呼んで，単なる区切りとして使っています．単なるといっても，区切りは不可欠なので，このような記号はむやみに省略しないでください．

3～5行をこのような形に書き直してみると下のようになります．

```
3  INPUT〝よこ〟；X
4  INPUT〝たて〟；Y
5  INPUT〝はんけい〟；R
```

具体的に3行を入力するときのキーの押し方を，まとめておきます．

カナのランプ光る　カナのランプ消える

3 スペース I N P U T（SHIFT ＋ 〃2）カナ よ こ カナ（SHIFT ＋ 〃2）； X
RETURN

このような修正をしたのちプログラムを実行すると，入力画面は次のようになります．

```
run
よこ? 58
たて? 89
はんけい? 20■
```

このように少しずつプログラムを修正してきた結果，中心と半径を指定して円を描かせるプログラムとしては，一応満足のいくものになったようです．MSXに触れながら実際に一つひとつ試してみれば，そう難しいことでもないと思います．

なお最後に，もう一度行番号を整理するために，[R][E][N][U][M] [RETURN] としておきましょう．

リスト

```
10 INPUT"よこ";X
20 INPUT"たて";Y
30 INPUT"はんけい";R
40 SCREEN 2
50 CIRCLE(X,Y),R
60 GOTO 60
```

COLUMN

キーボードについて（その1）

タイプライタやコンピュータの英文キー配列は，どの国のどの機種であっても同じになっています．この配列は，タイプライタが発明されてから何十年もかかって，英文を打つのに最も能率がよくなるよう改善した結果なのです．これに対しカタカナの配列は，いろいろな種類のものがあります．英文に比べカタカナは文字数も多く，どのような配列にしたら最も能率がよくなるのか，まだ結論が出ていないのです．

MSX BASIC 2-3 色を付けてみよう

これまでに説明してきた知識を使えば，好みの位置に好みの半径の円を描くことができます．次はこの円に色を付けてみることにしましょう．

MSXは，16の色を表示できるようになっています．これらの色にはそれぞれ0～15の番号が付いていて，赤，青，という代わりに，何番の色，というように指定します．MSXで使用できる色と，それぞれの色に対応する番号（カラーコードといいます）は次の表に挙げるとおりです．

カラーコード	色	カラーコード	色
0	透　明	8	赤
1	黒	9	明るい赤
2	緑	10	黄
3	明るい緑	11	明るい黄
4	暗い青	12	暗い緑
5	明るい青	13	紫
6	暗い赤	14	灰
7	水　色	15	白

カラーコードと色の対応

色の左に書いてある数字が，色を指定するときに使う番号です．

では，実際に色を指定して円を描いてみることにしましょう．ただし，今使っているディスプレイ装置がカラー表示できるものでないと，当然のことながら，カラーでは表示されませんので注意してください．

まずは文字の色を変えてみましょう．色を変える命令はCOLORです．

COLOR　10 RETURN

としてみてください．

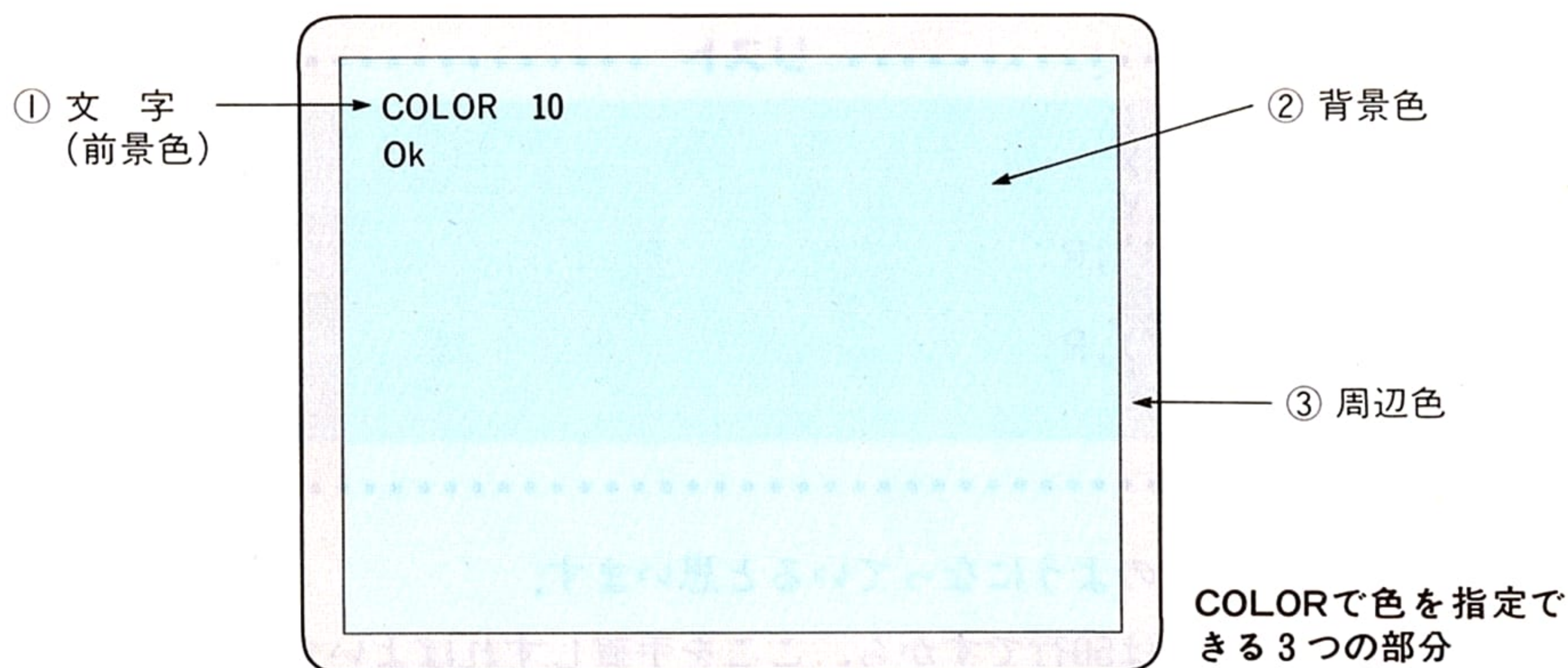
画面の文字の色が黄色に変わりました．同じように，COLOR　1とすれば文字が黒になります．COLOR命令はこのように，

COLOR　カラーコード

の型で使います．カラーコードは先の表にある0～15の番号です．

さて，もう一度ここで画面を見てみましょう．よく見ると，画面の色は①文字の部分，②背景の部分，③周辺の部分の3つに分かれています．

① 文　字
（前景色）
COLOR　10
Ok
② 背景色
③ 周辺色

COLORで色を指定できる3つの部分

今，文字はCOLOR　10で指定した色に，背景や周辺は電源を入れた時の色になっています．COLORを使えば，背景や周辺の色を変更することも可能です．

COLOR　文字の色，背景色，周辺色

の型で文字の部分，背景の部分，周辺の部分の色を指定できます．たとえば，

COLOR　15，1，1

とすると，文字は白くなり，背景や周辺は黒になります．色を使うときには，周りを黒

にしておくと，色がきれいに見えるので，背景や周辺の色を1(黒)にしておくとよいでしょう．

円に色を付ける

それでは，円に色を付けてみることにしましょう．色を付けるとき，あらかじめCOLOR命令を使ってもよいのですが，CIRCLEは命令の中で色を指定できるので，その方法でやってみます．

次のような型で使うと，CIRCLE命令で色を付けることができます．

CIRCLE (横，縦)，半径，色

色を0～15のカラーコードで指定することは，いうまでもありません．

2-2の最後に作ったプログラムを修正して，色の付いた円を表示させてみましょう．

リスト

```
10 INPUT"よこ";X
20 INPUT"たて";Y
30 INPUT"はんけい";R
40 SCREEN 2
50 CIRCLE(X,Y),R
60 GOTO 60
```

今，プログラムは上のようになっていると思います．

円を描いている部分は50行ですから，ここを手直しすればよいでしょう．ここで，

```
50 CIRCLE(X,Y),R,15
```

のようにカラーコードを直接指定してしまうと，色を変えるのに，いちいちプログラムそのものの修正が必要になります．そこで，カラーコードをCという変数で表して，次のようにします．

```
50 CIRCLE(X,Y),R,C
```

すると，円の位置や半径と同じように，キーボードからカラーコードを指定しなくてはなりません．そのために，30行と40行の間に次のような行を加えましょう．

```
35  INPUT "いろ"; C
```

以上の追加・修正が終わったらさっそく実行してください．自分の好きな色で円が描けるはずです．背景色と同じ色を指定してしまうと，何も見えませんので注意してください．もし，何も見えなくなってもあわてずに，CTRL キーを押しながら STOP キーを押してプログラムを終わらせ，もう一度別の色で試してみてください．

色を塗る命令

次にただ色のついた円を描くだけでなく，円に色を塗ってみましょう．

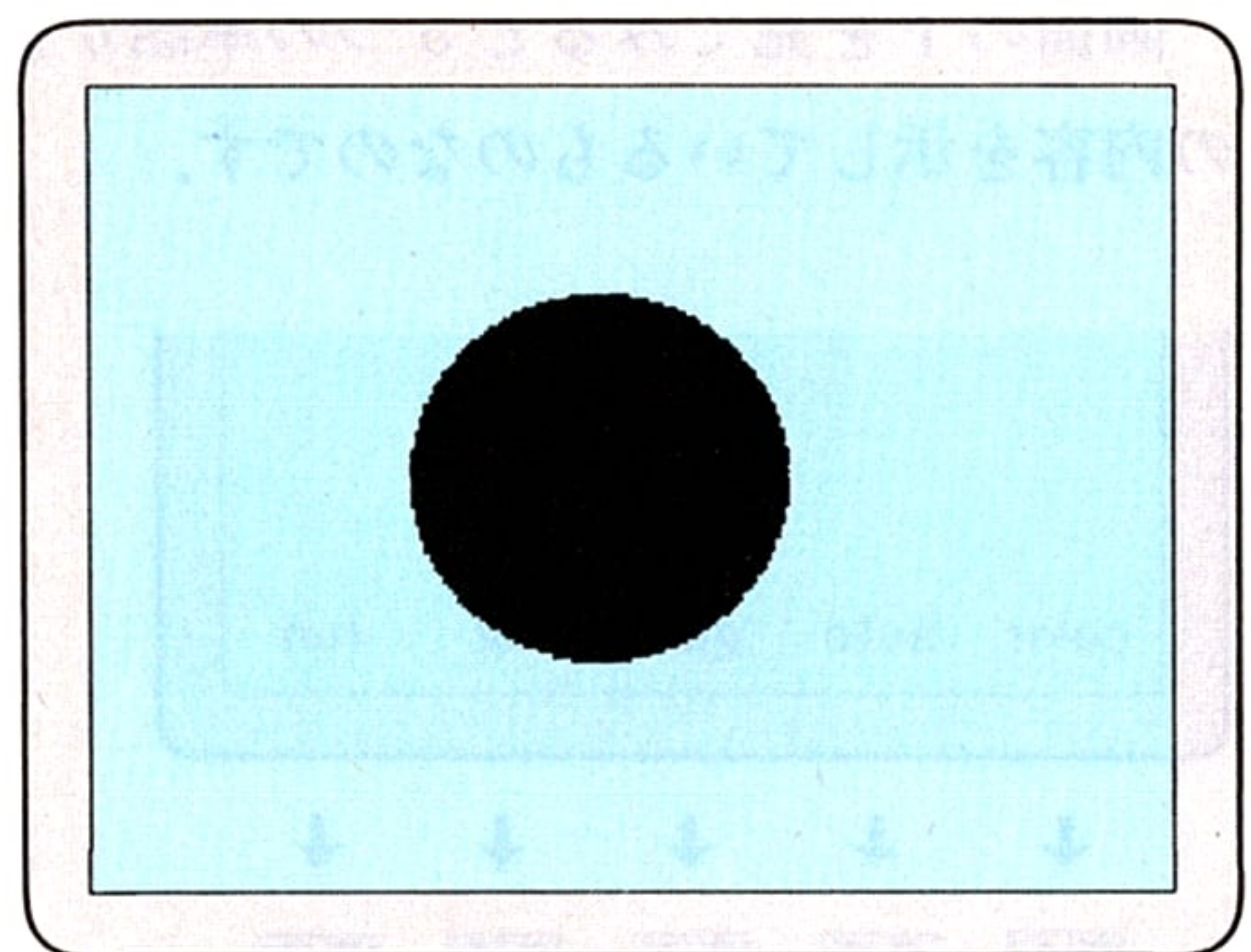
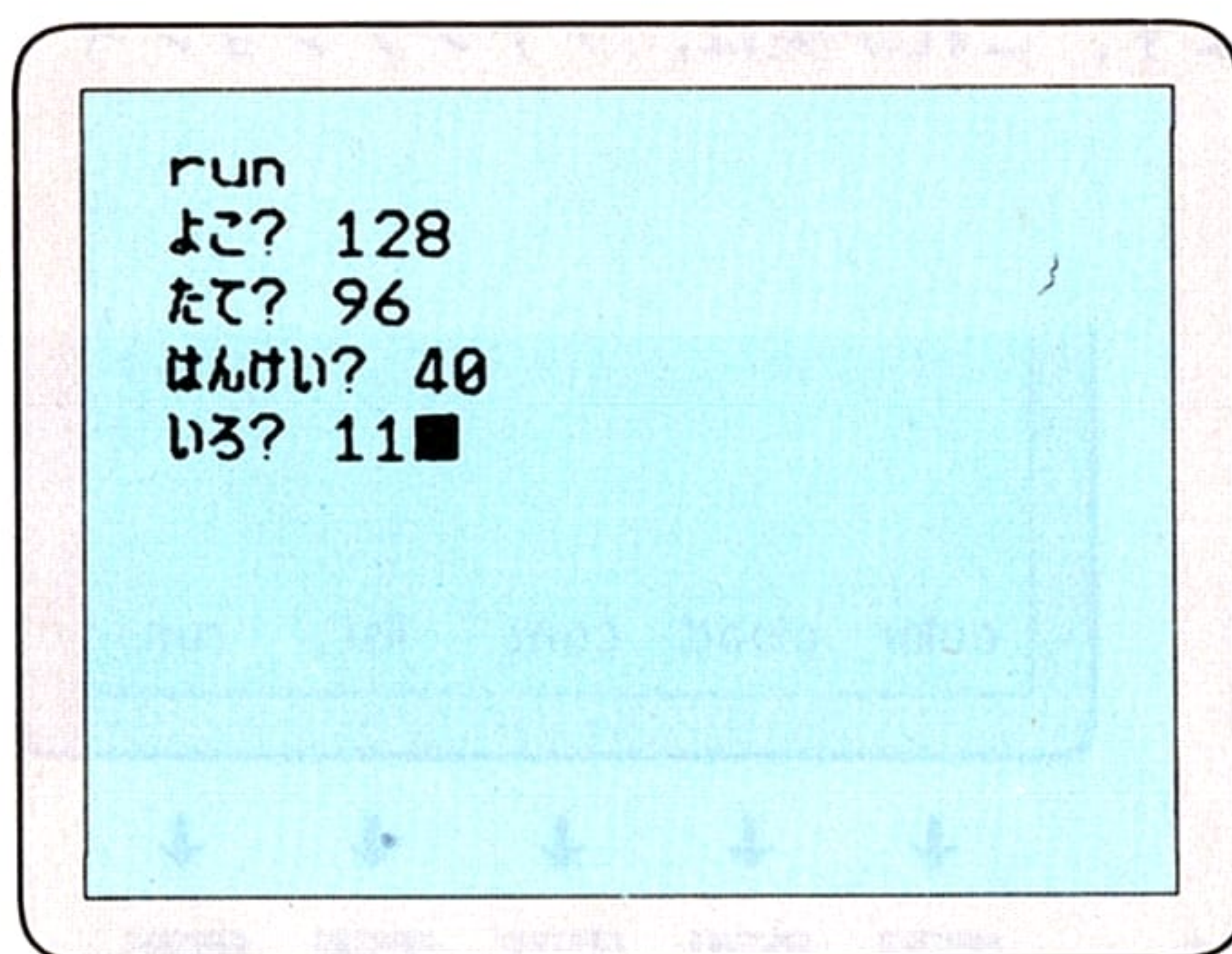
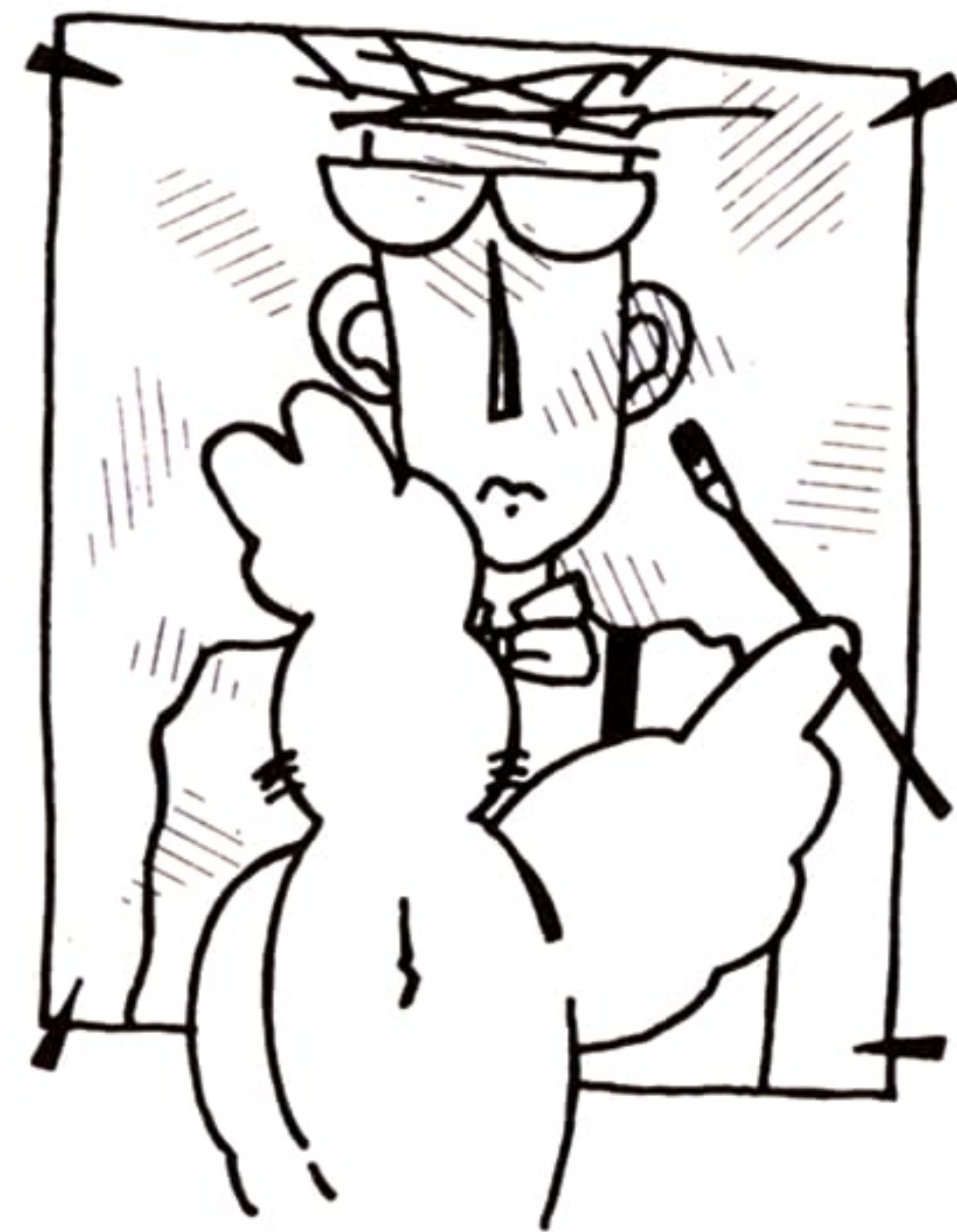
色を塗りつぶす命令は，PAINTです．この命令は次のような型で使います．

```
PAINT (横，縦)，ぬる色
```

横，縦は色を塗りたい場所の，どこでもいいから一点を指定します．言葉で説明してもよくわからないでしょうから，実際にやってみましょう．さきほどのプログラムに，

```
55  PAINT (X , Y) , C
```

という行を加えて実行してみましょう．

```
run
よこ? 128
たて? 96
はんけい? 40
いろ? 11■
```

ところで，円周の色と塗りつぶしたい色が違っていると，大変なことになります．興味のある人は，55行のCの代わりに15を指定してみてください．

今度は,

```
55　PAINT (0 , 0) , C
```

のようにしてください. 塗りつぶし始める点が, 円の境界の中にないと, 円の外側を塗りつぶしてしまうのです.

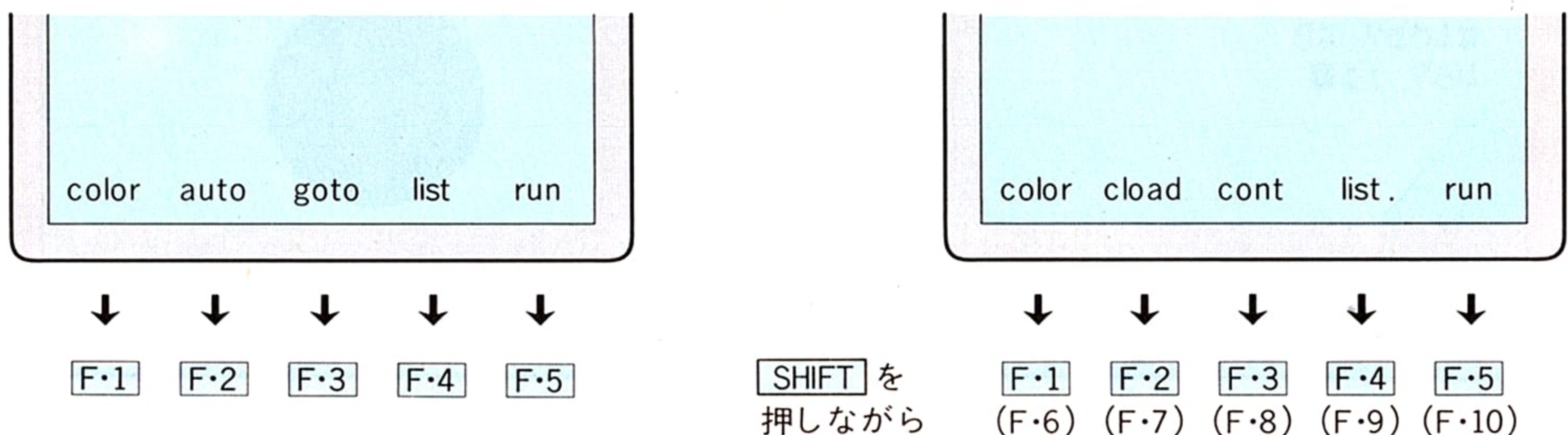
COLOR命令, PAINT命令を使えば, 16の色を自由にコントロールできるようになります. ここに挙げた例にとらわれず, 変数などを自由に変えて遊んでみてください.

便利なファンクションキー

さてこれまで, プログラムを実行するときや, リストを画面に表示させたりするときには, いちいち[R][U][N][RETURN]とか[L][I][S][T][RETURN]のように, 一つひとつキーを押してきました.

実はこのRUN, LISTなどは, キーボードの上の方に5つ並んでいるキー(ファンクションキーといいます)に登録されているのです. CHAPTER 1で説明ぬきに使ったのを覚えている人もいるでしょう. たとえば一番右の[F・5]のキーには, [R][U][N][RETURN]という文字が登録されているので, [F・5]という1つのキーを押すことで, プログラムが実行されます.

同じように[F・4]のキーにはLISTという文字が登録されているので, リストを取るためには, [F・4][RETURN]の2つのキーで済んでしまいます.

画面の下を見てみると5つの単語が並んでいます. これが実は, ファンクションキーの内容を示しているものなのです.

ファンクションキーと内容の表示

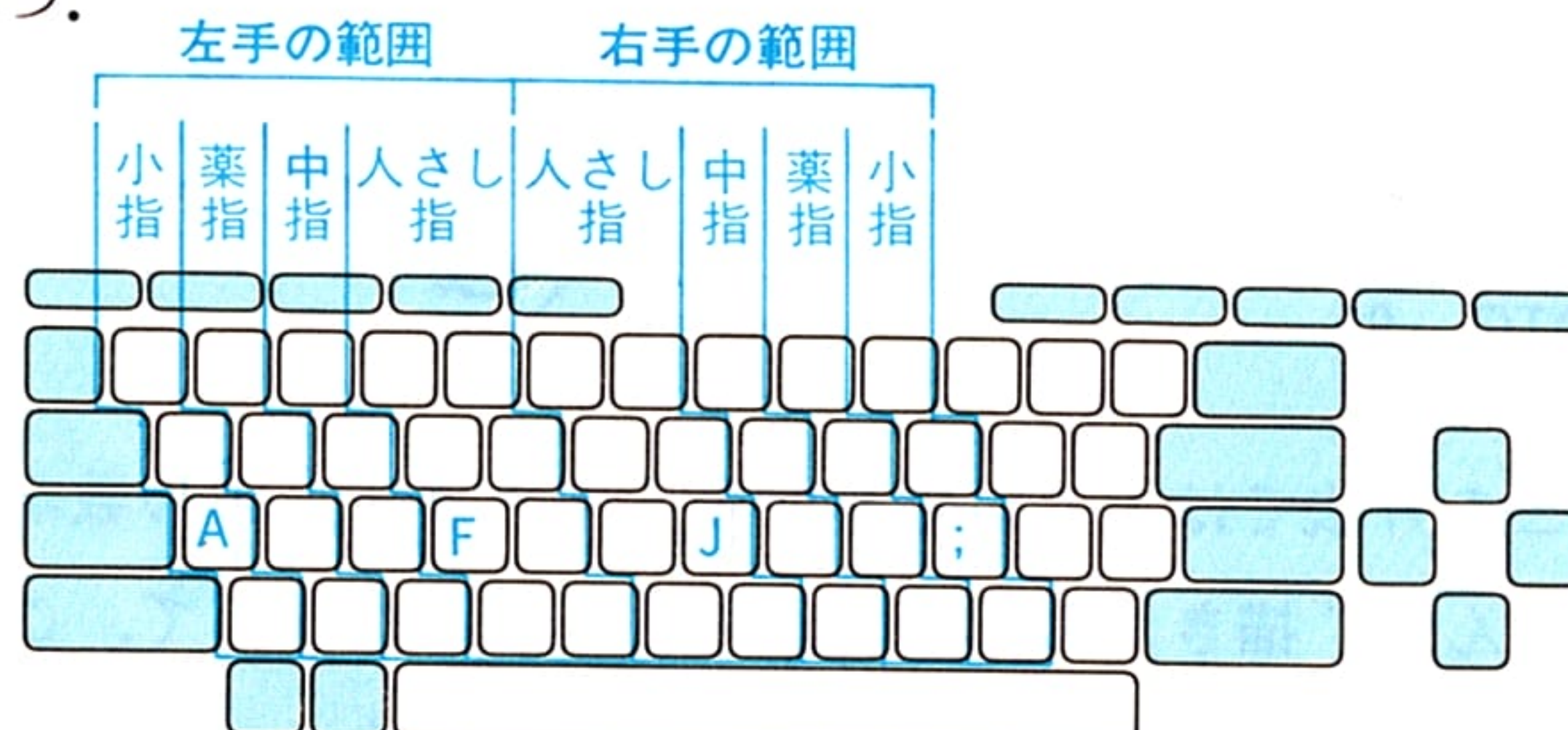
キーは5つしかありませんが，SHIFT キーと一緒に押すことによって，ここに登録されている10個の内容を使い分けることができます．それらをうまく使いこなせば，プログラムを打ち込んだり，修正したりするときの手間が省けるはずです．

ふだん画面に表れているものは，登録されている内容の一部です．F・1 ～ F・10 のすべての内容を表示させたいときには，KEY LIST 命令を使います．

```
key list
color
auto
goto
list
run
color 15,4,7
cload"
cont
list.
 run
Ok
■
```

COLUMN

キーボードについて［その2］

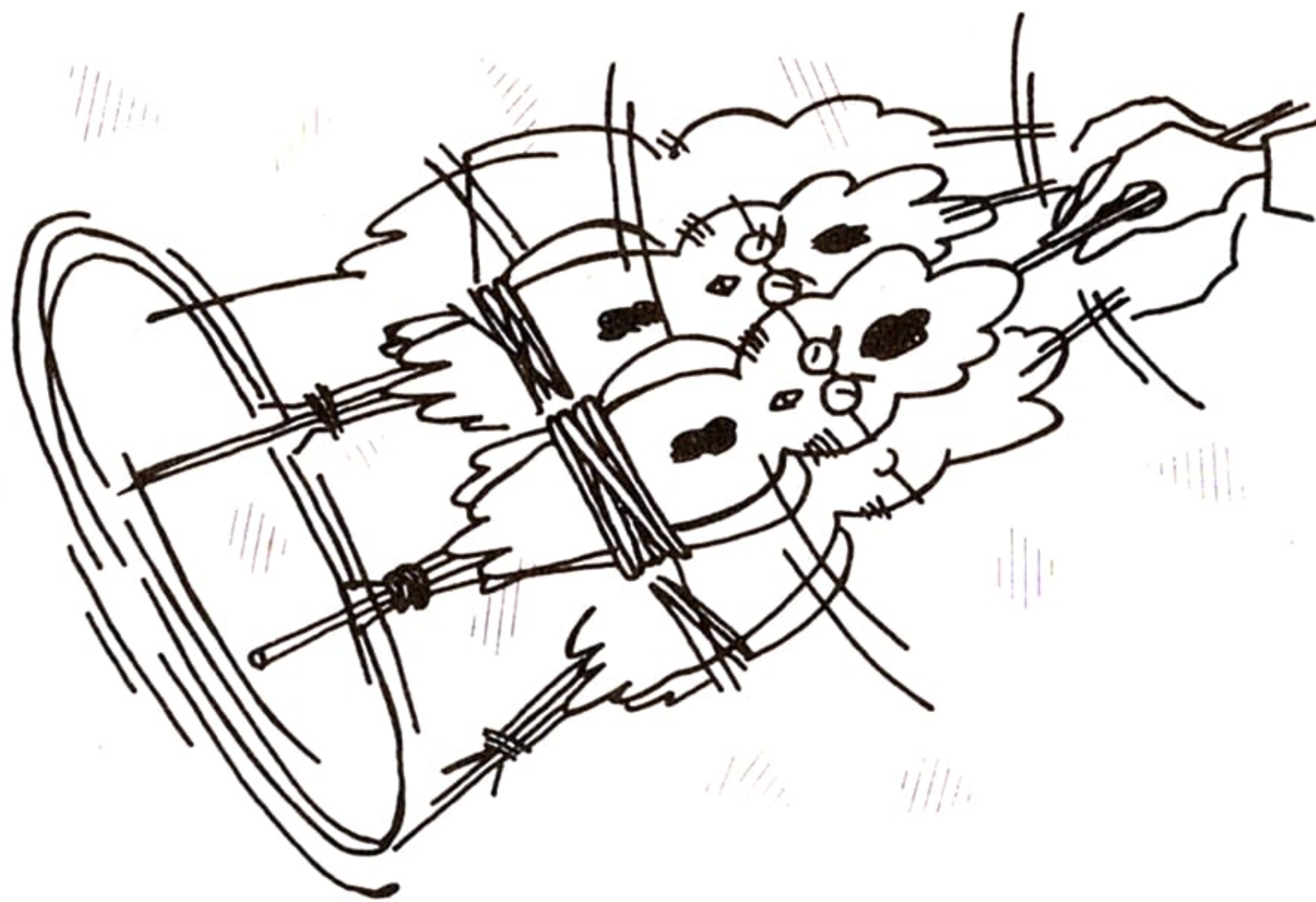
打ち込みを速く，楽にするには，指の置きかたがあるのです．下の図のとおりに指を置いて，それぞれの指で押すキーの範囲を忠実に守ってみましょう．

左手は小指から順にA,S,D,Fのキーに，右手は小指から順に;,L,K,Jの順に常に置くよう心がけます．

MSX BASIC 2-4 円によるデザイン

これまではいくら色を付けたりしても，しょせん1つの円を描いてみただけです．1つの画面に多くの円をきれいに並べて描けば，デザイン的にもう少しおもしろくなりそうです．

半径を変えよう

まずは，円の中心の位置は同じで，半径が少しずつ違った円を描いてみましょう．一番単純な方法として，次のようなものを思いつきます．

```
10  INPUT "よこ"; X
20  INPUT "たて"; Y
30  SCREEN  2
40  CIRCLE(X , Y) , 10
50  CIRCLE(X , Y) , 15
60  CIRCLE(X , Y) , 20
70  CIRCLE(X , Y) , 25
80  CIRCLE(X , Y) , 30
90  GOTO  90
```

しかし，この方法ではすぐに限界がみえてきます．円の数が少ないうちは別にこれでもかまいませんが，描きたい円の数が増えるにしたがって，この方法ではメンドウになります．何とか別の方法を考えなければなりません．

同心円をいくつか描くということを，もう少し整理して具体的にまとめると，次のようになるでしょう．

○ CIRCLE(X，Y)，Rで円を描く

○ Rを10から30まで5ずつ増やす

○ これを繰り返す

このような繰り返しによく使う命令が，FOR～NEXT～STEPです．

FOR～NEXT～STEPの使い方

FOR (変数) = [A] TO [B] STEP [C]

※（繰り返す内容）

NEXT (変数)

★(変数)の最初の値は[A]になる．※の部分を1回実行するごとに(変数)の値が[C]だけ増え，その値が[B]と等しいか，または小さければもう1回※の部分を実行する

★FORの後の(変数)とNEXTの後の(変数)は同じものでなければならない

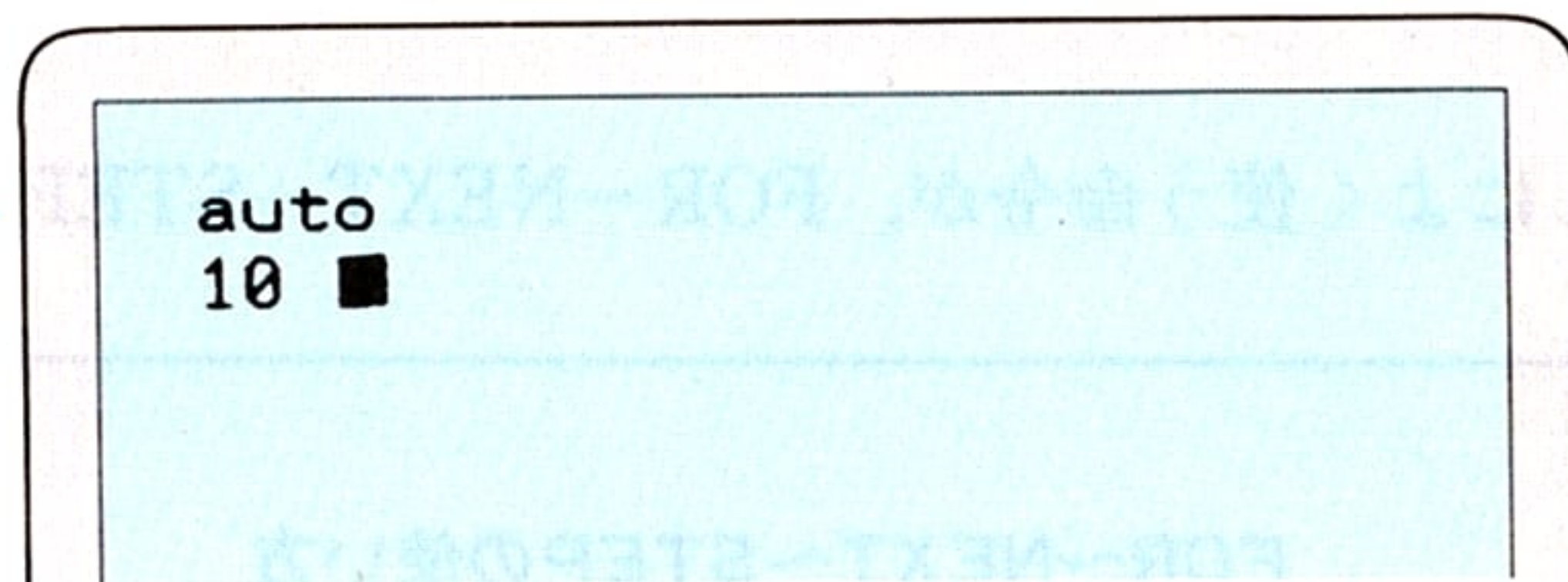
この命令は，FORからNEXTまでの一連の命令を，指定した回数だけ繰り返して実行するものです．どれだけ繰り返すのかを指定するのが，上のA，B，Cで，変数をAからBまで，Cずつ増やしながら繰り返します．たとえばFOR I=1 TO 10 STEP 3となっていれば，Iの値が1，4，7，10と増えて，計4回，NEXTで囲まれた範囲の命令を繰り返すことになります．

具体例がないとわかりにくいでしょうから，半径を変えて円を描くことを例にして，FOR～NEXTの使い方を見ていきましょう．

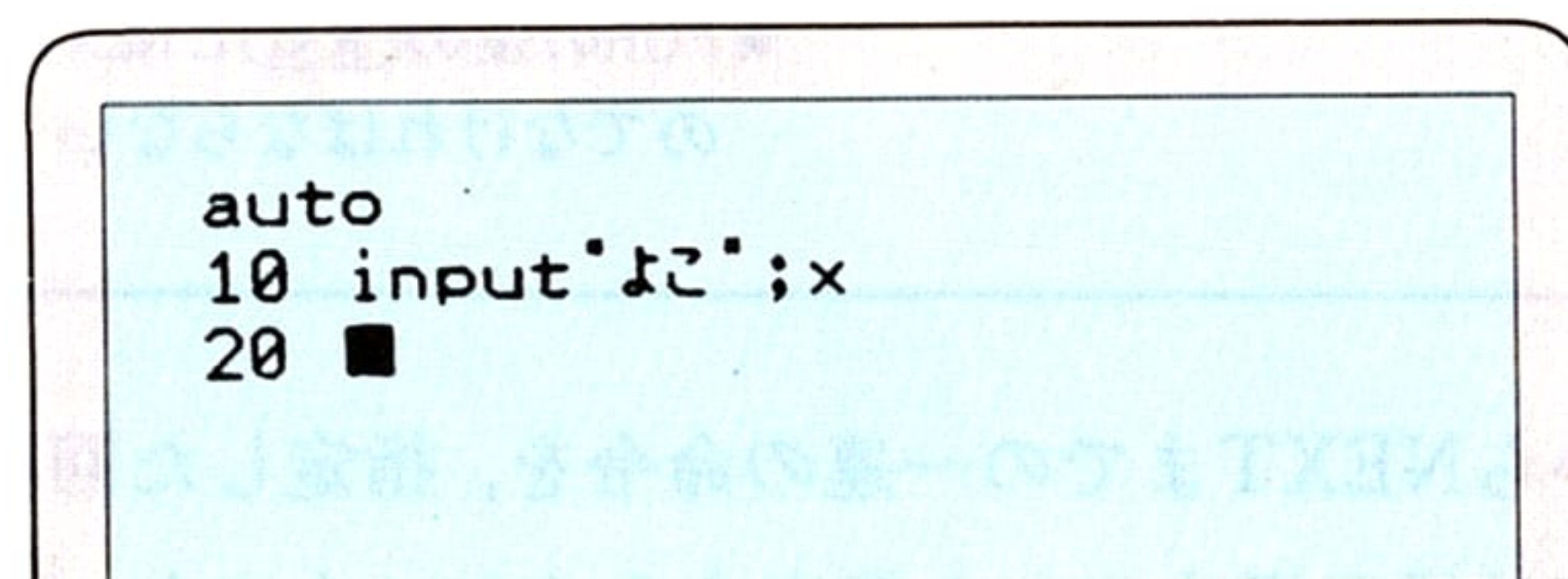
リスト

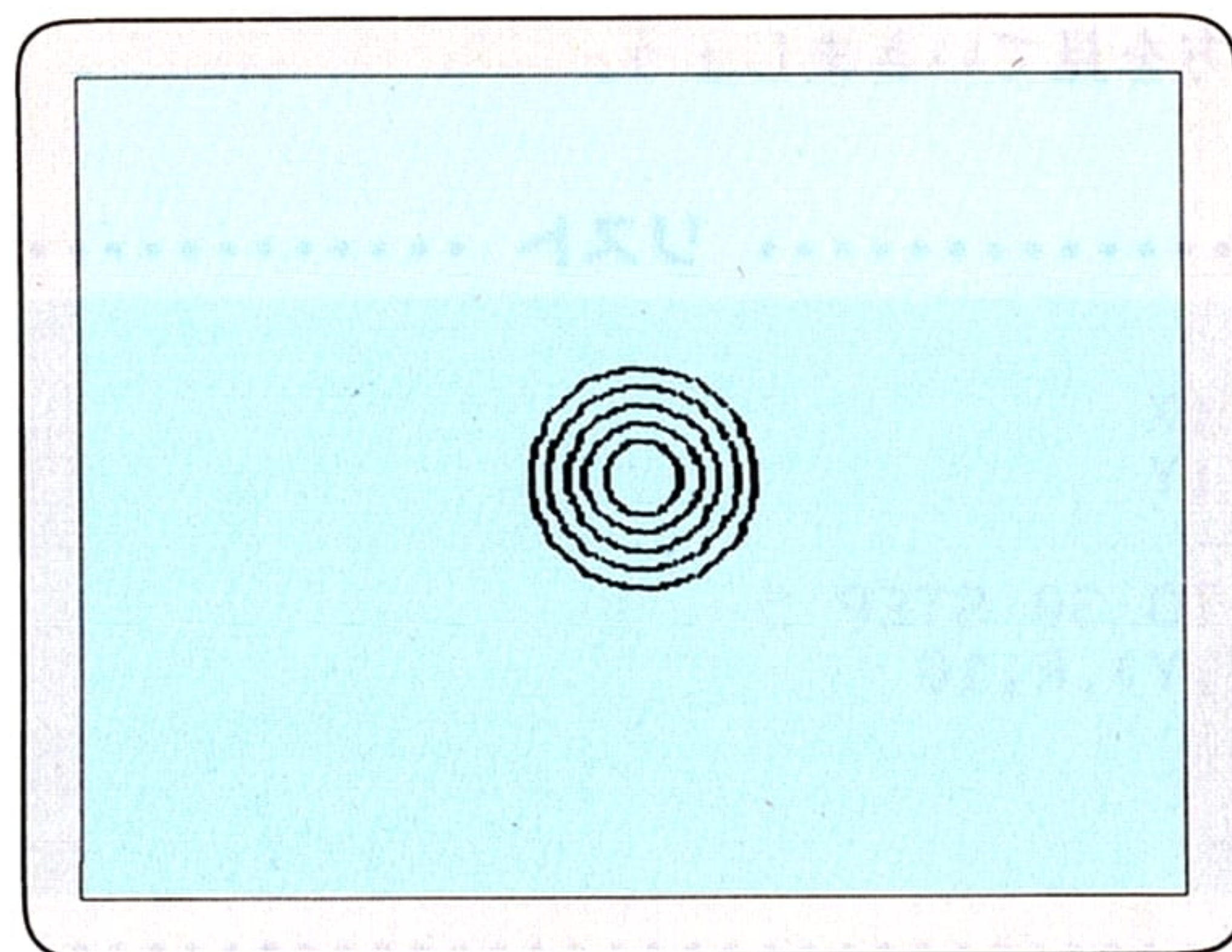
```
10 INPUT"よこ";X
20 INPUT"たて";Y
30 SCREEN 2
40 FOR R=10 TO 30 STEP 5
50  CIRCLE(X,Y),R,10
60 NEXT R
70 GOTO 70
```

まずこのプログラムを打ち込まなければなりません．ここで，行番号の入力の手間を楽にする命令を紹介しましょう．AUTO RETURN と打ち込んでください．

```
auto
10 ■
```

画面には，autoという文字と，10という行番号が表れました．AUTOを使うと，そのつど行番号を打ち込まなくてもよいので，プログラムの入力が楽になります．

行番号が出たら，いきなりINPUT･･･とその行の内容を打ち込みます．1行入力し終わって RETURN を押すと，すかさず次の行番号が表れます．

```
auto
10 input"よこ";x
20 ■
```

以下，同じようにプログラムを打ち込んでいきます．プログラムを打ち込み終わって，行番号の自動発生を止めたいときは，CTRL キーを押しながら STOP キーを押します．

さて，プログラムの打ち込みが終わったら実行してみましょう．

目的の同心円が描けました．この結果を見てもわかるとおり，ここではFOR～NEXT～STEPを使って，半径Rの値を10から30まで，5ずつ変えながら中心点の同じ円を描いています．

FORの後の変数と，NEXTの後の変数は，必ず一致させなければなりません．

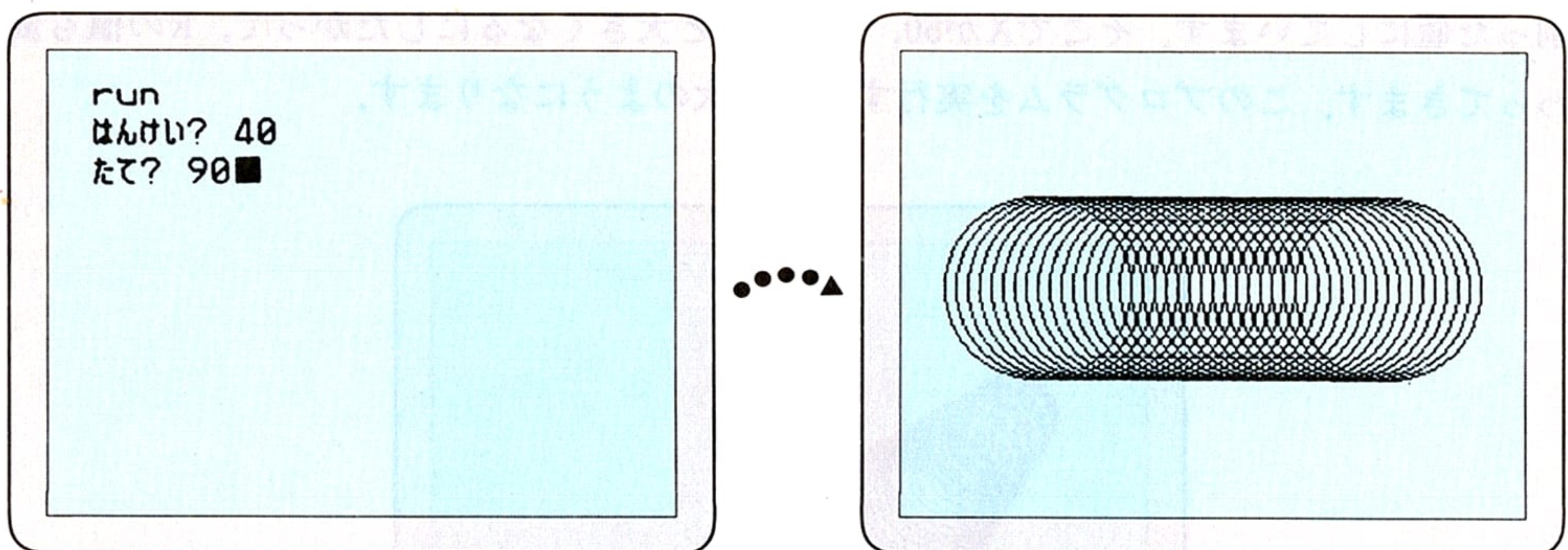
位置を変えよう

今度は，別の例題で，FOR～NEXT～STEPの機能を確かめてみましょう．半径を一定にして，円を描く位置を変えて，いくつも並べて描いてみます．今のプログラムを修正して次のようにしてください．

リスト

```
10 INPUT"はんけい";R
20 INPUT"たて";Y
30 SCREEN 2
40 FOR X=50 TO 200 STEP 5
50  CIRCLE(X,Y),R,10
60 NEXT X
70 GOTO 70
```

修正したプログラムを実行すると，次のようになります．

```
run
はんけい? 40
たて? 90■
```

10行，20行のINPUT命令で，いろいろな値を入力して試すだけでなく，40行のFOR～NEXT～STEPの中で指定してある，50，200，5などの数字を変えてみるのもおもしろいでしょう．

こういった値を変えるとき，カーソルを動かして修正するだけでなく，FOR～NEXT～STEPの中の値を変数に代えてももちろんかまいません．

```
25  INPUT "step"; s
```

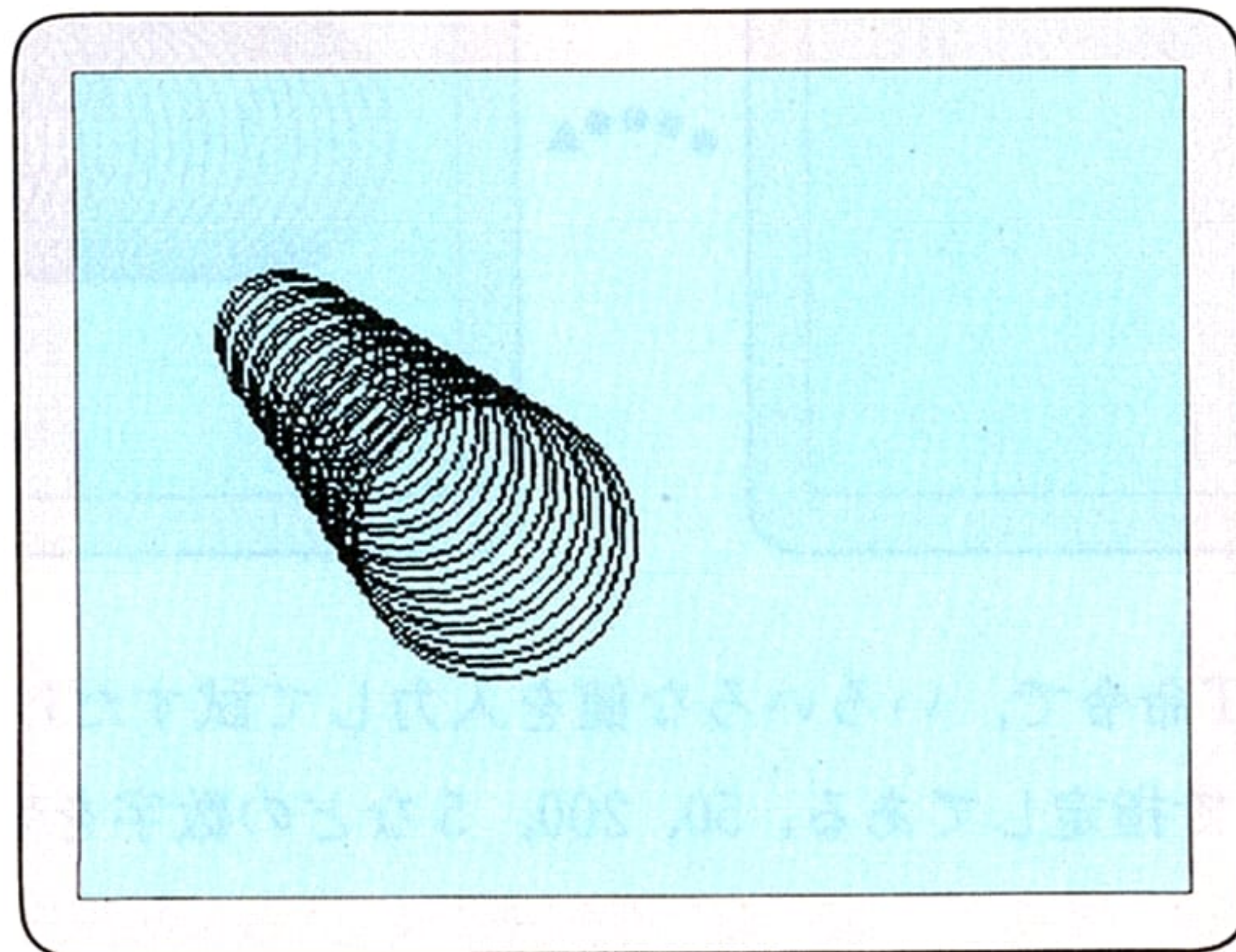
を加えて，40行の STEP 5を，STEP Sにすれば，円を描く間隔も自由に設定できるようになります．興味のある人は，さらに色の指定を加えたりするのもよいでしょう．

円を使ったデザイン

次のプログラムを見てください．

リスト

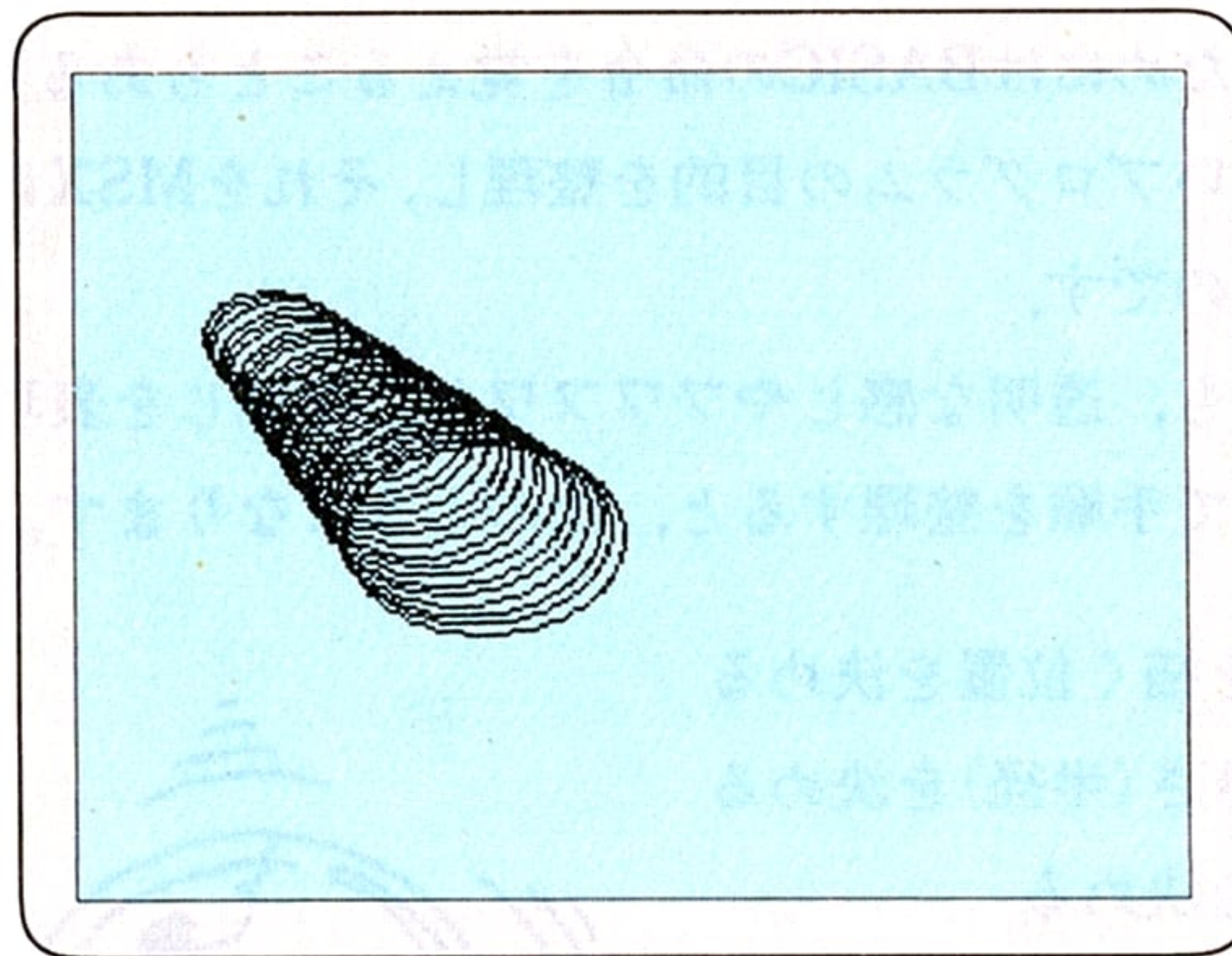
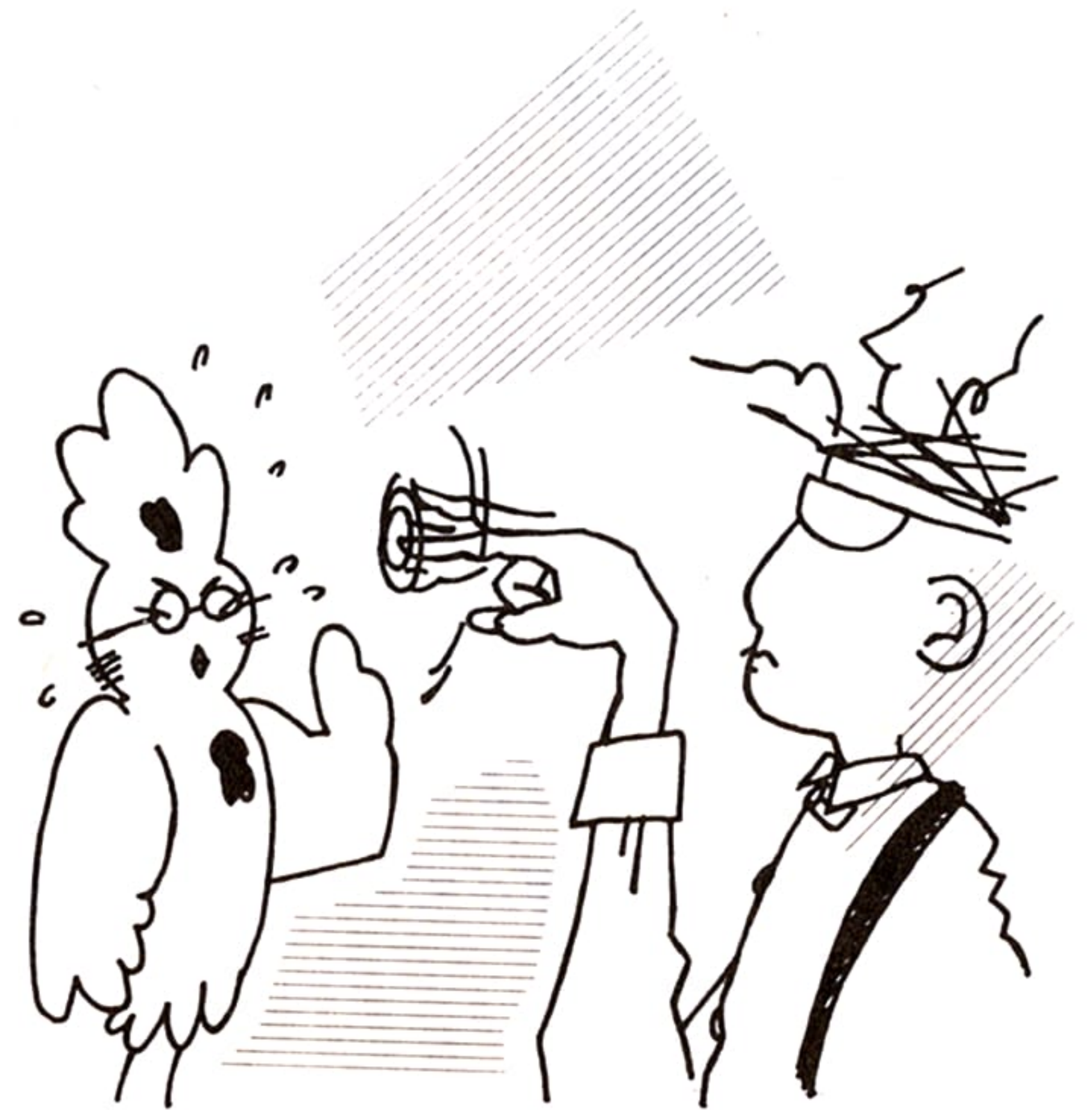
```
10 SCREEN 2
20 FOR X=50 TO 100 STEP 2
30  R=X/3
40  CIRCLE(X,X),R,10
50 NEXT X
60 GOTO 60
```

このプログラムでは，位置も半径も変えていきながら円を重ねて描いていきます．30行で R＝X／3 というのが出てきます．これはR＝X÷3の意味で，半径は常にXを3で割った値にしています．そこでXが50，52，54，と大きくなるにしたがって，Rの値も変わってきます．このプログラムを実行すると，次のようになります．

ずいぶんデザインらしくなってきました.

なお上のプログラムで，30行，40行の行番号と命令の間が1文字分あけてあるのに気付いた人もいると思います．プログラムの実行上特に意味はありませんが，こうしておくと，FOR～NEXT～STEPで繰り返す範囲がはっきりして，リストが見やすくなるのです．

さて，40行をちょっと直して下のようにしてください．

```
40  CIRCLE (X , X) , R , 10 , , , , .7
```

先ほどのプログラムに［,,,,.7］が付いているだけです．これを実行してみると下のようになります．

［,,,,.7］の，［.7］は円の比率を表しています．比率が1以下のときは横長の楕円，1以上ならば縦長の楕円になります．ここでは，0.7を指定しているので横長の楕円になります．この比率をいろいろと変えてプログラムを実行してみてください．なお，特にこの比率を指定しないときは，比率は1とみなされます(つまり普通の円になるわけです)．

円を描くプログラムも，ここまでくればほぼ仕上っています．

MSX BASIC

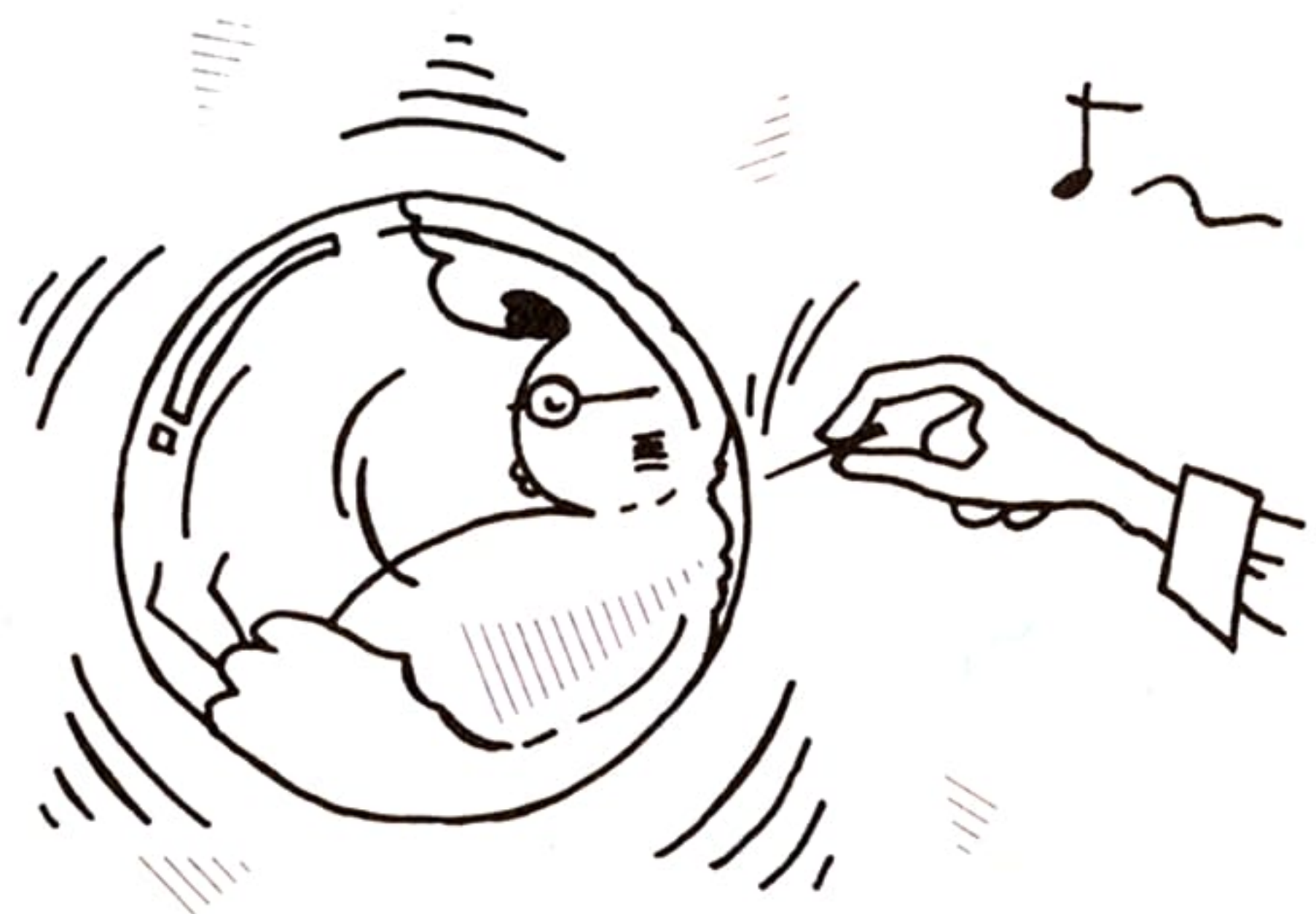
2-5 しゃぼん玉とばそう

これまでのまとめとして，円の描きかた，色の塗りかた，繰り返し，などをすべて使ったプログラムを作ってみましょう．目標は，カラーのしゃぼん玉を画面に飛ばすプログラムです．

手順の整理

しゃぼん玉を飛ばすプログラムを作る前に，どういう手順でしゃぼん玉を描くことができるかを整理してみましょう．プログラムを作るときには，問題の整理が一番大切です．プログラムを作るためにはBASICの命令を覚えることもある程度必要です．が，それ以上に自分の作りたいプログラムの目的を整理し，それをMSXに実現させる手順を考えることの方が重要なのです．

しゃぼん玉といっても，透明な感じやフワフワとした感じを表現するのは難しそうです．今の実力を考慮して手順を整理すると，次のようになります．

① しゃぼん玉(円)を描く位置を決める
② しゃぼん玉の大きさ(半径)を決める
③ しゃぼん玉の色を決める
④ しゃぼん玉を描く
⑤ 描いたしゃぼん玉に色を塗る
⑥ 以上の手順を繰り返して，画面のあちこちにいろいろな大きさ，いろいろな色の円(しゃぼん玉)を繰り返しいくつも描いていく．

色を塗る部分

さて，以上のような手順に基づいて，プログラムを作ってみます．

まず，いままでの知識でできるのは，円を描く部分と，色を塗る部分ですから，さきにこの部分を打ち込んでおきましょう．これまでと同じように，横をX，縦をY，半径をR，色をCという変数を使って表すと，

```
70 CIRCLE (X, Y), R, C
80 PAINT (X, Y), C
```

のようになります．行番号が10から始まっていませんが，これは後で行を追加する予定があるからです．同じようにしておいてください．

さて，上の2つの行で，X，Y，R，Cの値が決まれば，円を描いて色を塗ることができます．X，Y，R，Cの値をどのように指定したらよいでしょうか．

しゃぼん玉は作るたびに大きさも位置もいろいろで，規則性はありません．また色もひとつだけでなく，規則性をなくした方がよいでしょう．

毎回値を変える，しかもそこに規則性があってはまずいということで，これまでのようなやり方(たとえば2-4のようにFOR～NEXT～STEPで値をかえていく)ではうまくいきません．

RND命令

このようなとき，普通でしたらサイコロを使いますが，実はMSXの中にもサイコロが用意されているのです．本物のサイコロには1～6の6つの目しかありませんが，コンピュータのサイコロは，出る目の数を6つどころか100,200といったように設定することができます．

この機能を利用して，あたかもコンピュータの中でサイコロを振らせるようにして，X，Y，R，Cの値を決めてやればよいのです．

MSXの中にあるサイコロ，つまりランダムな数値を作り出す機能をもつ命令が，RNDです．

RNDは，たいがいの場合次のような型で使います．ひな形として覚えておくとよいでしょう．

RNDを使って整数の乱数を作る方法

RND(1)で０以上１未満の小数の乱数を発生する．このままでは使いにくいので次のような型で，整数の乱数にして使う

(1) A＝INT（RND（1）＊ 整数値）

　Aに０～(整数値 － 1)の範囲の整数の乱数が入る

(2) A＝INT（RND（1）＊ 整数値＋1）

　Aに１～ 整数値 の範囲の整数の乱数が入る

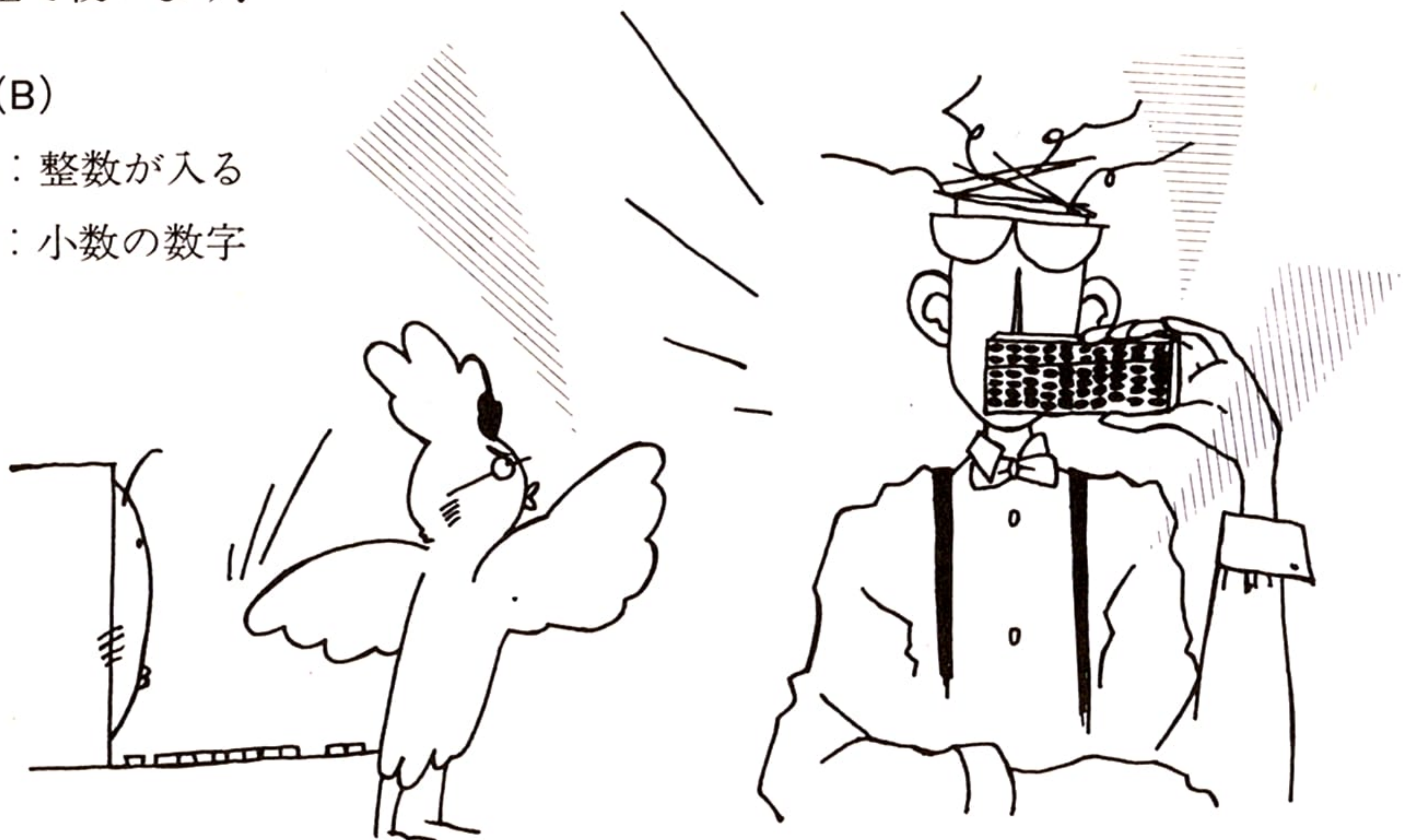
ここでRNDのほかに，INTとか［＊］とか見慣れないものがいくつか出ていますので，これらについて解説しておくことにしましょう．

RNDそのものでは，０～１の間の，小数の数字を作り出すだけなのです．これを０～10などの範囲にするためには，RNDで作られた値を10倍なり20倍なりしてやらなければなりません．このかけ算を行っているのが，［＊］です．普通のかけ算で使う×マークは，Xと間違いやすいのでコンピュータでは使われません．

RNDで得られた値が，たとえば0.59521943994623だったとしましょう．０～10の範囲の乱数が欲しいときは，これを10倍にします．そうすると5.9521943994623になります．さらにこれを整数にしたいときは，INTを使います．INTは小数を整数にする命令で，次のような型で使います．

A＝INT(B)

　A：整数が入る

　B：小数の数字

いろいろな円

これら，RNDやINTを使ってコンピュータでのサイコロを作り，X，Y，R，Cなどの値を決めてみましょう．

絵を描くための画面は，横256，縦192の細かい点から成り立っています．縦横それぞれ0～191，0～255の番号が付いていて，この数字で位置を指定することは前から説明しているとおりです．

横Xの値を0～255の間の整数の中から選びます．

```
30　X＝INT (RND ( 1 ) ＊256)
```

同様に縦の位置を0～191の範囲で選びます．

```
40　Y＝INT (RND ( 1 ) ＊192)
```

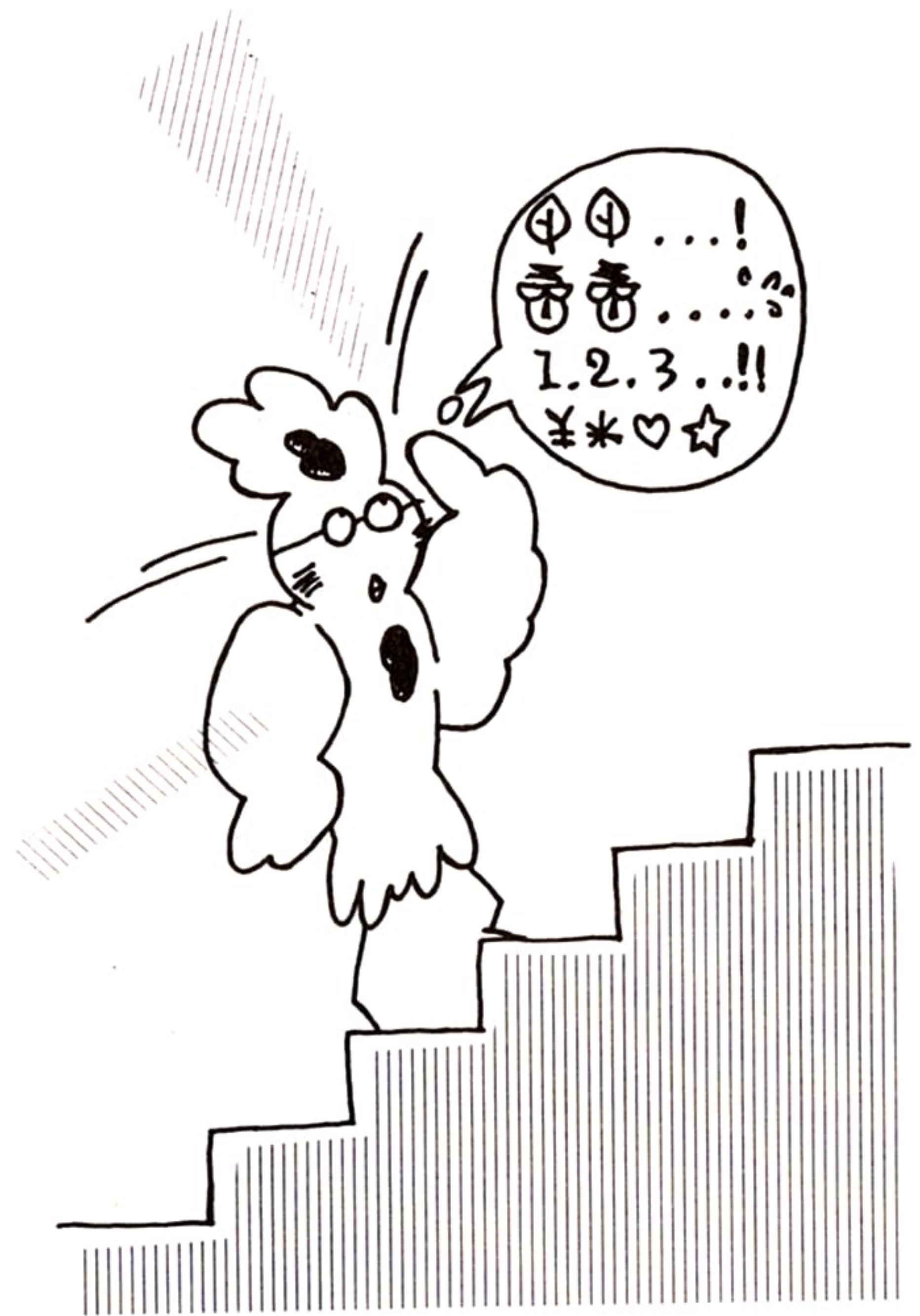
次に半径の大きさを指定します．半径がゼロというと円ではなくなってしまうので，1～30の間から半径の大きさを指定するようにしています．

```
50　R＝INT (RND ( 1 ) ＊30＋1)
```

色はカラーコード1から15の間で選びます．

```
60　C＝INT (RND ( 1 ) ＊15＋1)
```

これで位置，半径，色をランダムに変えて円を描き，色を塗ることができるようになりました．

あとは，これまでのプロセスを何回でも繰り返せばよいのです．繰り返しにFOR～NEXTを使ってもよいのですが，この方法だと繰り返す回数は有限回になってしまいます．さまざまな色，大きさのしゃぼん玉をずっと描き続けられるようにはできないものでしょうか．

何度も繰り返す

ここで，いままでのプログラムの最後に付いていた，GOTOという命令を思い出してください．これまでGOTOはおまじないとして使っていましたが，この命令は，実はこのような繰り返しに役にたつ命令なのです．

GOTO命令は指定した行へ飛ぶ命令です．今の例では 90 GOTO 30 という命令を付け加えると，90行になると再び30行から実行されるようになるのです．これで何度でも無限に繰り返しが行われるようになります．

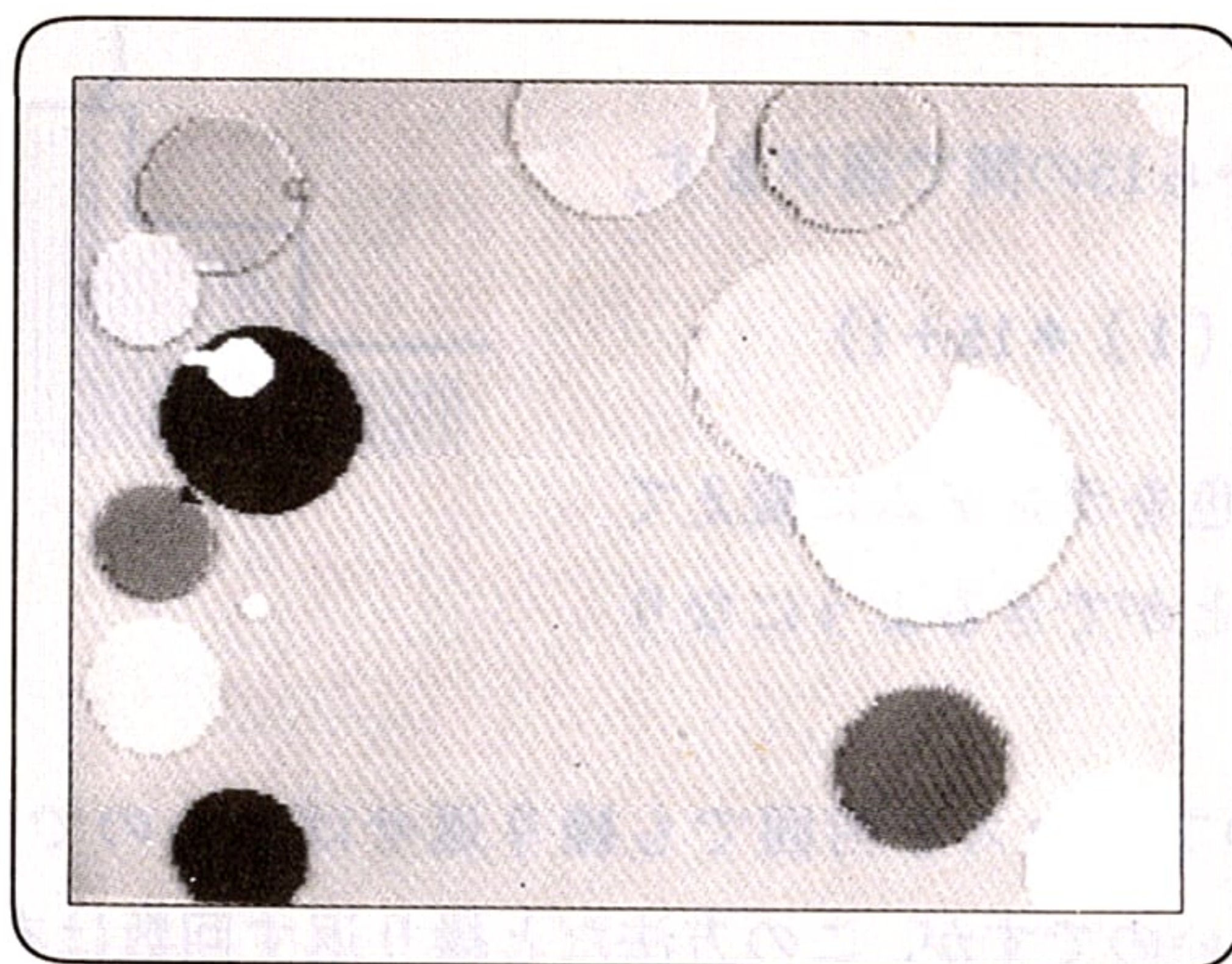
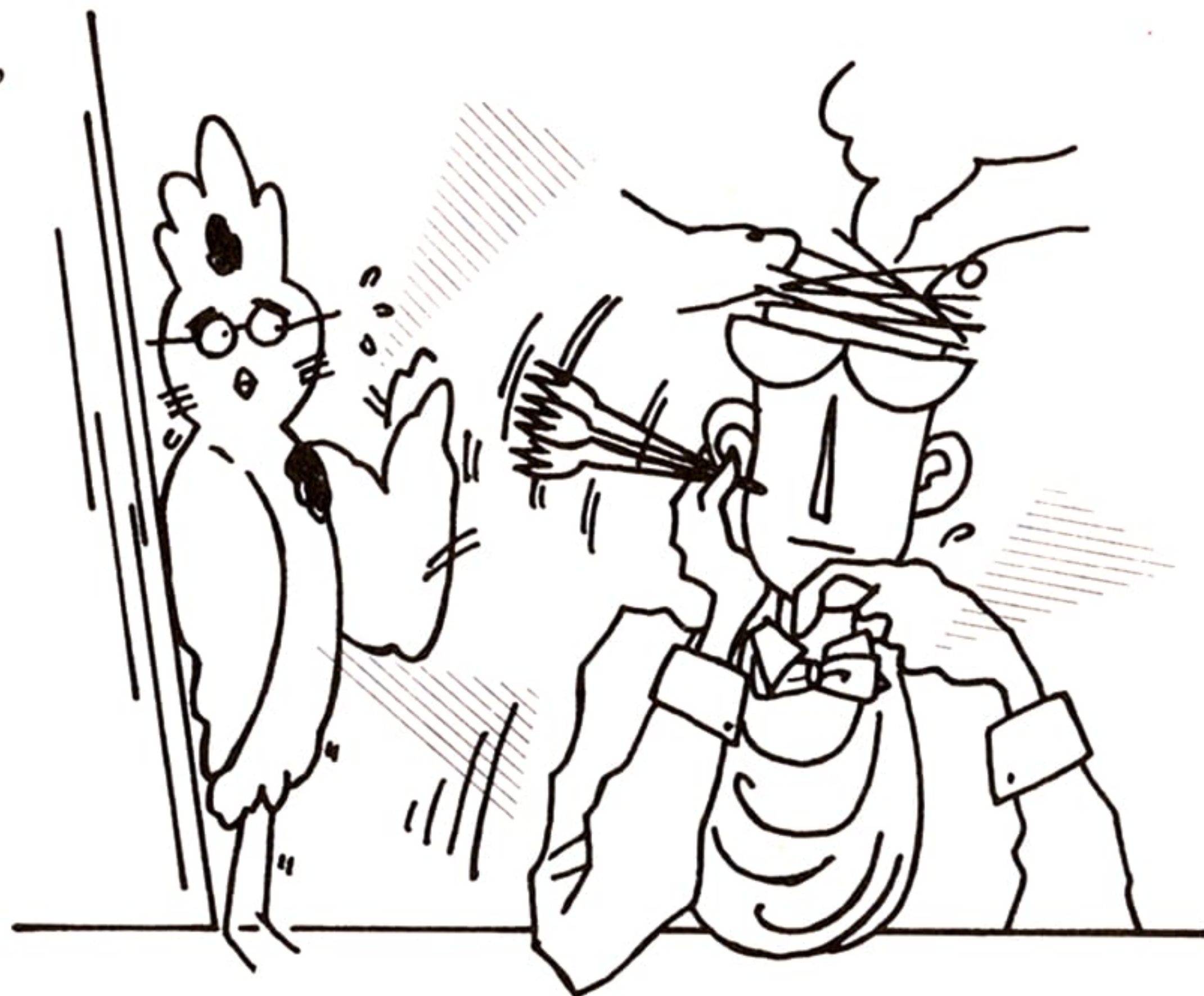
ここまでのリストに絵を描く準備である，

```
20 SCREEN 2
```

を付け加えてください．

実行の順序は20→30→40→50→60→70→80→90→30→40→……となります．

ここまでプログラムができたら，実際に試してみましょう．

画面のあちらこちらに，いろいろな円が表示され続けます．

これを止めるには CTRL キーを押しながら STOP キーを押します．

乱数の順序を変える

何回か試してみるとわかるとおり，実行するたびに同じパターンで表示されます．コンピュータは非常に正確なので，サイコロの振り方も前回と寸分違わずに振ってしまうのです．これを毎回変えるためには，次のような行を加えます．

```
10 X=RND(-TIME)
```

こうすると,毎回同じパターンになることはありません.RNDの中がマイナスの型はまだ説明しませんが,この行をプログラムの頭の方に入れておくようにしてください.

これでプログラムは一応完成です．下にリストを載せておきます．

リスト

```
10 X=RND(-TIME)
20 SCREEN 2
30 X=INT(RND(1)*256)
40 Y=INT(RND(1)*192)
50 R=INT(RND(1)*30+1)
60 C=INT(RND(1)*15+1)
70 CIRCLE(X,Y),R,C
80 PAINT(X,Y),C
90 GOTO 30
```

描かせる円の大きさを変えたいときは，50行の30とあるところを変えます．半径を小さくしたいときは 20,10 などに，もっと大きくしたいときには 40,50 などの値にすればよいでしょう．ただし，あまり半径が大きいと，画面いっぱいひとつの円で塗りつぶすような感じになってしまい，あまりきれいではありません．

TIMEについて

このプログラムでちょっと出てきたTIMEは，時間を表す特別な変数になっています．電源を入れた時をゼロとして，1秒間に60ずつカウントしています．ゲームのプログラムなどで，時間を計ったりするのによく使われます．この値は，

```
TIME=数字
```

の形で自由に設定できます．

MSX BASIC

2-6 たいせつなプログラムの保存

ここまで「円を描く」という課題をいろいろと発展させながら，プログラムを作ってきました．そろそろ，プログラムを保存する方法を覚えておくとよい頃でしょう．

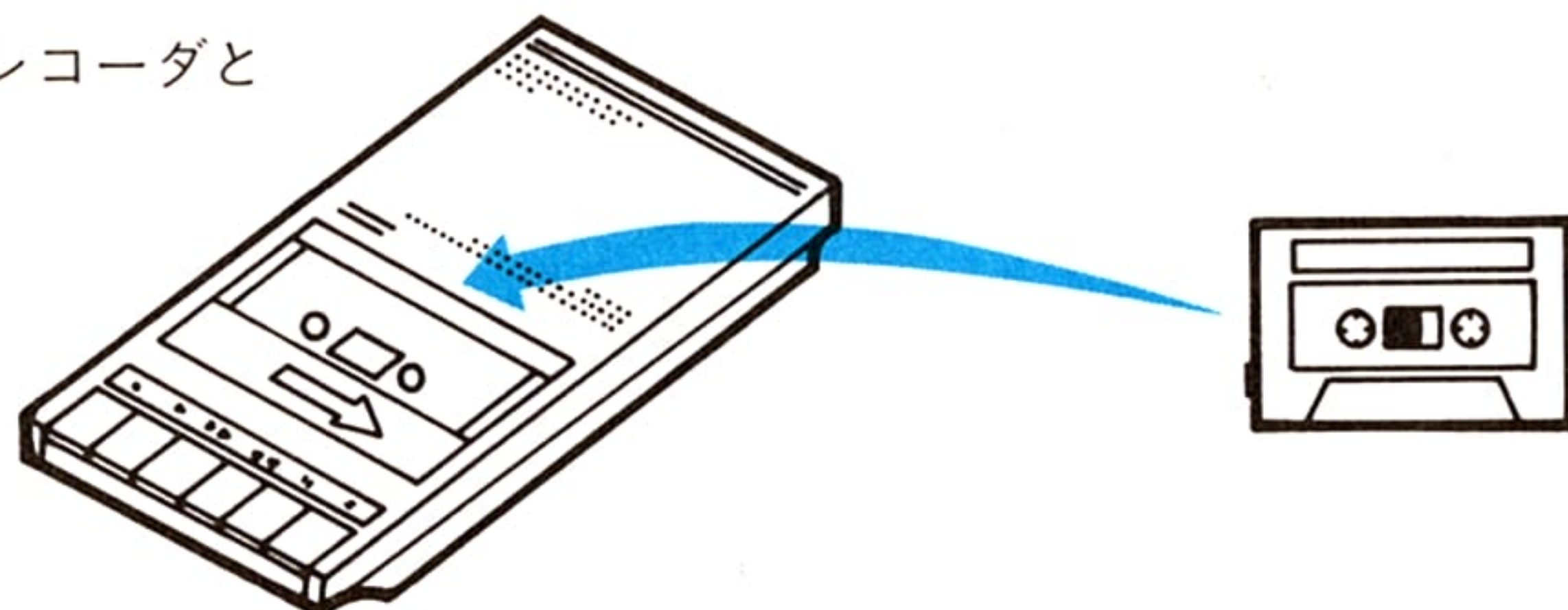
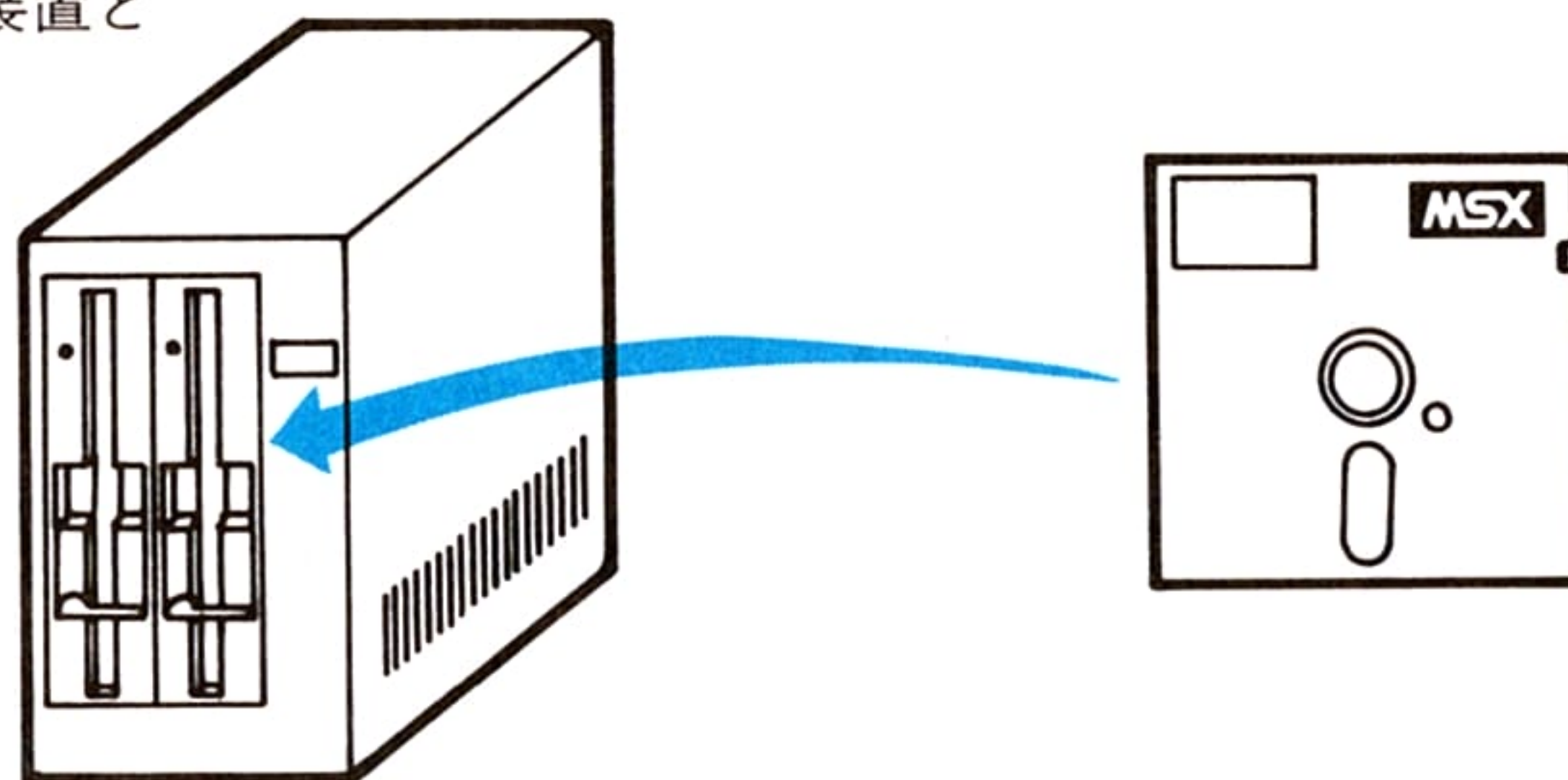
コンピュータの記憶装置(メモリ)は，一部のハンドヘルドコンピュータやポケットコンピュータを除いて，電源が入っているときにはきちんとプログラムやデータを記憶していますが，ひとたび電源スイッチを切ると，すべてを忘れてしまいます．「絶対に電源を切らなければいいのだろう」と考える人がいるかもしれません．しかし，自分の意図とは無関係に停電することもあるでしょうし，間違って電源コードに足をひっかけてしまうこともあるでしょう．

それだけでなく，コンピュータが一度に記憶していられるプログラムの本数は，特に工夫しないかぎりは１本です．１台のMSXに常に１種類だけのプログラムを入れておくならともかく，普通はそうはいきません．

いずれにせよ大きなプログラムを作るようになれば，プログラムをコンピュータの記憶装置の外に，別に保存しておくことが必要になります．

MSXの外部記憶装置

コンピュータ本体の外でプログラムやデータを蓄える装置を，外部記憶装置といいます．MSXで使える外部記憶装置とその特徴を，下にまとめてみましょう．

カセットテープレコーダと
カセットテープ

フロッピーディスク装置と
フロッピーディスク

MSXで使える主な外部記憶装置とその媒体

○ 磁気テープ装置

普通のカセットテープレコーダのことです．普段わたしたちが音楽を聞くときに使っているテープに，プログラムやデータを記憶します．なにしろ身近にある機械で，かつ安価だという理由から，パーソナル・コンピュータの外部記憶装置として広く使われています．ただし，情報の読み書きに時間がかかるのが欠点です．

○ フロッピーディスク装置

カセットテープレコーダについで多く使われている外部記憶装置です．値段が高いのが欠点ですが，プログラムやデータの読み書きが速く，かつ正確に行われるというメリットがあります．MSXでは，ディスクを使うために，MSX DISK BASICとMSX-DOSが用意される予定です．これらについては，また別の機会に詳しく説明したいと思います．

ここでは，身近にあって安価なカセットテープにプログラムを保存してみましょう．すでにCHAPTER 1でカセットテープからプログラムを読み込んでゲームをさせてみましたから，MSXとカセットテープレコーダのケーブルのつなぎ方は，わかっているでしょう．それでは，プログラムをテープに保存する方法を見ていきたいと思います．

プログラムを保存する

プログラムをカセットテープに記録する命令は，CSAVEです．CSAVEは下のような型で使います．

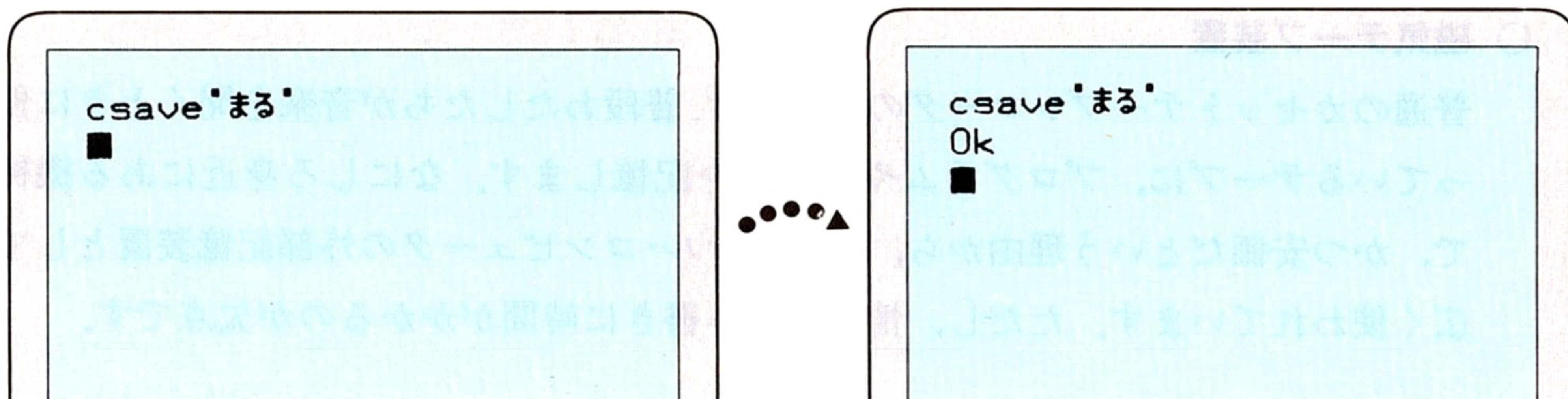
CSAVE〝プログラム名〟

CSAVE命令を使うときは，記録しておくプログラムに名前を付けておきます．この名前は6文字以内で指定します．ここでは〝まる〟という名前を付けて，さきほどのしゃぼん玉プログラムをカセットに記録(保存)してみましょう．

① カセットを録音状態にする
② CSAVE〝まる〟 RETURN と打ち込む

カチッと音がしてテープが回り始めます．しばらくするとOkの表示が出てカセットが止まり，プログラムの保存は完了します．

```
csave"まる"
■
```

```
csave"まる"
Ok
■
```

このとき注意することは，当たり前ですが，正しくカセットを入れ，リーダーテープの部分を送ってから記録することです．カセットは普通の音楽用のものでもかまいません．あまり録音時間の長いものよりも，短いものの方が便利でしょう．最近ではマイコン用と銘打って，C-20(片面の録音時間が10分)などのテープも売られていますので，それらを使うのもよいでしょう．

保存したプログラムの確認

以上の手続きでプログラムがテープに保存されます．しかし，CSAVE命令でテープに保存しても，100パーセント正確に保存されているとは限りません．

そこで，正確にプログラムが保存されているか確認してみましょう．今保存したばかりのプログラムを再びテープから読み込んで，本体の記憶装置に残っているプログラムと照合してみればよいのです．このための命令が，CLOAD？です．この命令は，

CLOAD？

または，

CLOAD？〝プログラム名〞

の型で使います．プログラム名を省略したときには，最初に読み込まれてきたプログラムと，MSX本体の中のプログラムとの照合を行います．

まず，カセットを巻き戻してください．

巻戻しのボタンを押してもテープが回らないときには，

MOTOR [RETURN]

としてください．カチッと音がして巻戻しができるはずです．

次に，

CLOAD？[RETURN]

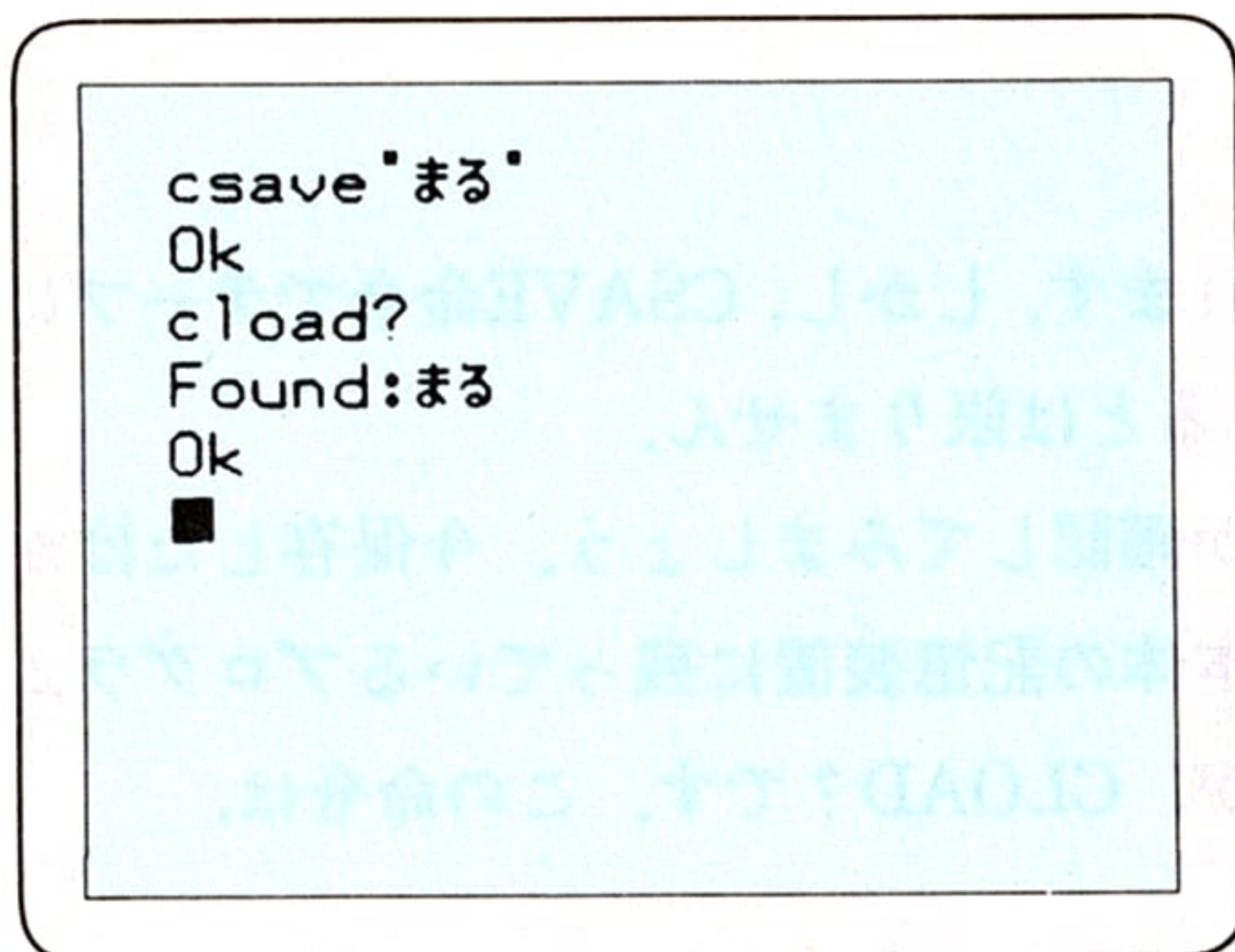
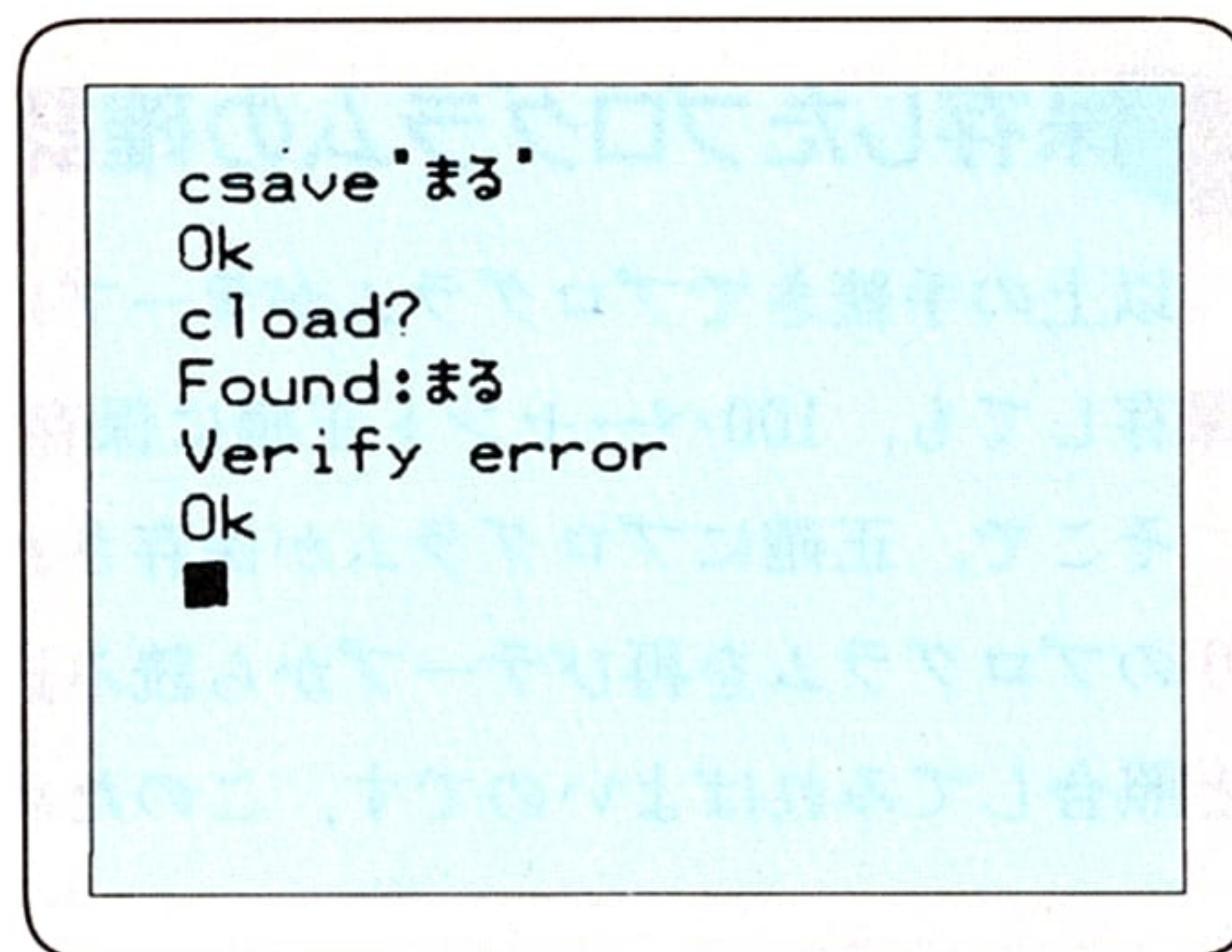
として，カセットを再生状態にしてください．少し待っていると，

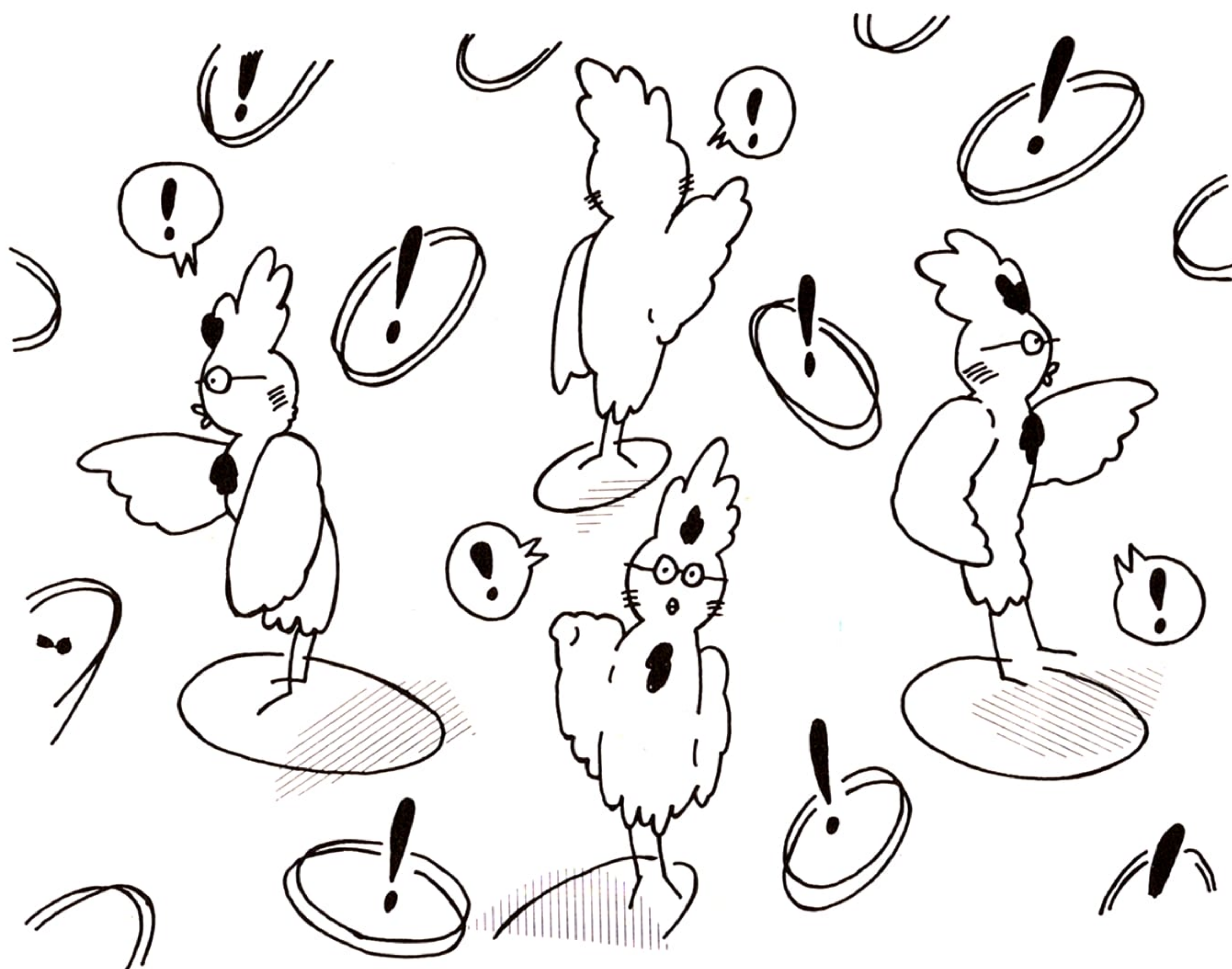
Found：まる

のように表示されます．照合がうまくいけばそのあとOkが表示されますが，うまくカセットに保存されていないときは，Verify　error と表示されます．こうなったときは，もう一度 CSAVE 命令で保存をやり直してください．大丈夫だと思っていても，いろいろな落とし穴が潜んでいて，プログラムの保存がうまくいかないことがあります．CSAVE命令のあとは，CLOAD？命令で必ず確認(照合)作業を行うようにしましょう．

```
csave"まる"
Ok
cload?
Found:まる
Ok
■
```

照合がうまくいった場合

```
csave"まる"
Ok
cload?
Found:まる
Verify error
Ok
■
```

照合に失敗した場合

うまく読み込みができないときは，カセットのボリュームを調整してみてください．安物のテープを使ったり，ボロボロのカセットテープレコーダを使うと，エラーが多発することが多いようです．

プログラムの読込み

ここまでうまくいったら，確実にプログラムがテープに保存されたと考えてよいでしょう．安心して電源スイッチを切ることができます．実際に電源を切って，保存したばかりのプログラムを読み込んでみましょう．

テープからプログラムを読み込む命令は，1-4でも説明したとおり，CLOADです．CLOAD命令はCLOAD？命令と，使い方もひな形もほとんど同じです．ただし，ひとたびCLOADで新たなプログラムを読み込むと，MSX本体の記憶装置の中にあったプログラムは消えてしまいますので注意してください．

では，今カセットに記録したばかりのプログラムを，CLOAD命令を使って読み込んでみましょう．

① **カセットを巻き戻す**
② **CLOAD** RETURN
③ **再生ボタンを押す**

以上の操作で，さきほどカセットに保存したプログラムが読み込まれます．

```
cload
Found:まる
Ok
list
10 X=RND(-TIME)
20 SCREEN 2
30 X=INT(RND(1)*256)
40 Y=INT(RND(1)*192)
50 R=INT(RND(1)*30+1)
60 C=INT(RND(1)*15+1)
70 CIRCLE(X,Y),R,C
80 PAINT(X,Y),C
90 GOTO 30
Ok
■
```

ひとつのカセットテープに多くのプログラムを保存したときは，プログラム名を指定して読み込むとよいでしょう．

あなた自身のプログラムライブラリが充実してくれば楽しいですね．

COLUMN

フロッピーに負けないカセットテクニック

カセットテープは安価で手軽な外部記憶装置ですが，フロッピーディスクなどに比べると，かなり不満も感じさせます．カセットに対する不満とは，大別すれば次の3つにつきるのではないでしょうか．

(1)よくエラーをおこす　(2)操作がめんどくさい　(3)読み書きが遅い

これらのうち(1)(2)は工夫しだいでほとんど気にならない程度まで解決することができます．その方法のいくつかを紹介しましょう．

[その1]　エラー対策

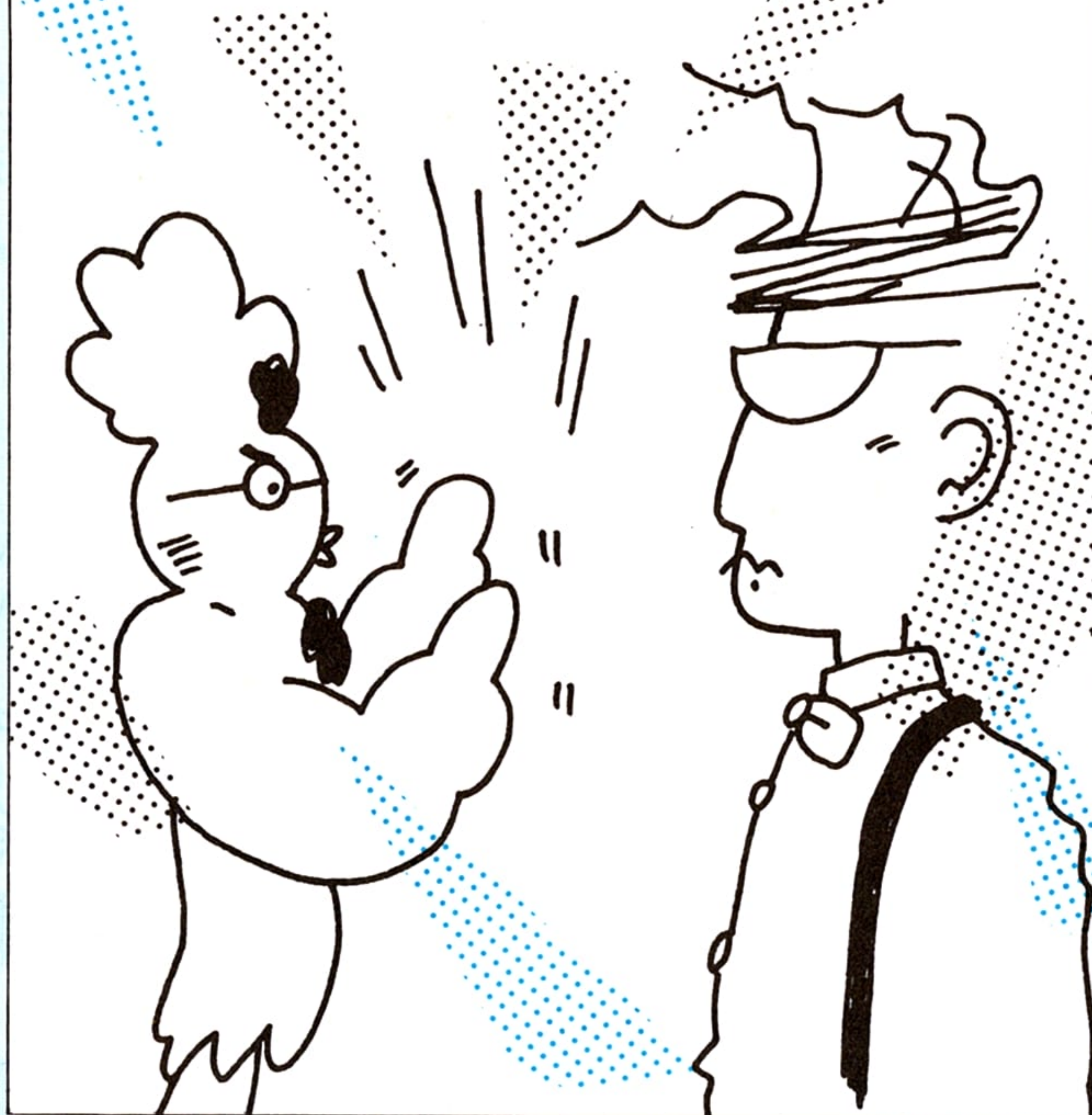
- ●テレコのヘッドを掃除する．やり方がわからない人はオーディオ好きな友達に頼むか，テレコに付いてくる取扱説明書を見る．
- ●ボリュームのレベルはシビアに決める．マニュアルで勧める音量にこだわらずあれこれ変えてみる．ボリュームの合う範囲は思っていたよりずっと狭いことが多い．
- ●必ず新品のテープを使う．安売りの詰め替え品は避ける．
- ●カセットデッキやステレオラジカセを使う場合は，LかRの片チャンネルだけに接続する．両方に接続するとエラーが多発する．

[その2]　操作性の向上

- ●カセットテープは，マイコン専用のものを使う．リーダー部分(テープの頭の録音できない部分)がないためテープの頭からすぐ記録でき，音楽用テープのようにリーダー部分にうっかり記録してしまう失敗が起きない．また，テープの時間が短いので，プログラムの整理がしやすい．
- ●カセットケースには，プログラム名と，そのプログラムがテープのどのへんの位置にあるかがわかるように，テープカウンタの数字をメモする習慣をつける．
- ●マイクロカセットも使うことができる．しかし，エラーは起きやすくなるので，テープは最高級のものを使う．

CHAPTER 3

楽しくプログラミング

——スロットマシン，グラフ，暗号——

MSX BASIC 3-1 スロットマシンを作ろう

CHAPTER 2 で円を描いて遊んでいるうちに，プログラムというものがだんだんわかってきたことと思います．BASICの命令も，かなり覚えました．CHAPTER 3 ではさらに進んで，何か目的をもった，もう少しまとまったプログラムを作っていくことにしましょう．最初の目標は，スロットマシンゲームです．

ゲームの設計

ただ漠然と，「MSXでスロットマシンゲームを作ろう」と思っても，なかなか自分が想像しているとおりには作れません．最初にどのようなゲームにするかを具体的に決めておくことが必要です．あまり欲ばったことを考えても，計画だおれになる心配があります．ここでは次のように決めましょう．

○ 最初に持ち点が1000点ある
○ 持ち点の範囲内で点を賭ける．もし，持ち点より多く賭けようとしても，できないようにする
○ 2つの数字を乱数で出す
○ もし2つの数字が一致したら，賭け点は2倍になって戻ってくる
○ そうでなければ，賭け点は没収される
○ 持ち点があれば再び賭ける
○ 持ち点がなくなればゲームはおしまい

プログラムは，次のようになります．

リスト

```
100 'するっとましん
110 X=RND(-TIME)
120 M=1000
130 PRINT"もちてん=";M
140 INPUT"かけてん=";K
150 IF K>M THEN 130
160 A=INT(RND(1)*10)
170 B=INT(RND(1)*10)
180 PRINT
190 PRINT A;B
200 PRINT
210 IF A=B THEN M=M+2*K:GOTO 230
220 M=M-K
230 IF M>0 THEN 130
240 PRINT"おしまい"
250 END
```

①準備の部分（100〜120）
②賭ける部分（130〜150）
③数字の表示部分（160〜200）
④持ち点の計算の部分（210〜220）
⑤持ち点チェックの部分（230〜250）

このプログラムは大きく分けて，①準備の部分，②点を賭ける部分，③数字を表示する部分，④当たりはずれの計算，⑤持ち点のチェック（終わりの判定）の5つの部分から成っています．まだ知らないBASICの命令も含まれていますので，順を追って説明していきましょう．

準備の部分

準備の部分では，乱数の準備，最初の持ち点の設定，といった作業を行います．

行番号は0〜65529の間で付けられるので，100から始めることにしましょう．こうしておくと行番号が増えてもだいたい3ケタで収まり，リストが見やすくなります．

```
100 'するっとましん
```

さて，100行には，行の頭にアポストロフィ［'］が付いて，日常使っている言葉がそのまま書いてあります．［'］はBASICのプログラムの中でプログラムのタイトル，製作者の名前，製作年月日などをメモしておきたいときに使います．

次に乱数の準備を行います．ここでは，2-5と同じようにRNDを使います．

```
110 X=RND(-TIME)
```

このあと最初の持ち点を決めれば，準備の部分は完了です．持ち点は，Mという変数で表し，1000点とします．

```
120  M=1000
```

点を賭ける部分

点を賭ける部分を，もう少し細かく整理してみると，（1）持ち点を表示する，（2）賭け点をキーボードから入れる，（3）もし，賭け点が持ち点より大きいときは（1）に戻りもう一度賭け点を入れさせる，というように分けることができます．この（1）（2）（3）のプロセスをプログラムにしてみるとどうなるでしょうか？

まず（1）です．

画面に文字や数字を表示する命令はPRINTです．この命令は下のような型で使います．

PRINTの使い方

(1) PRINT 変数……………………………変数の内容を表示する

例） PRINT A

(2) PRINT "メッセージ"…………………メッセージを表示する

例） PRINT "MSX"

(3) PRINT "メッセージ" ; 変数………メッセージと変数を横に並べて同じ行に表示する

例） PRINT "きんがく" ; K

★[;]は区切りの記号．[;]で区切ると同じ行にいくつもの文字や数字を表示することができる

★区切りの記号として[,]も使うことができる．[;]で区切るとすぐ横に並べて表示されるが[,]の場合は間隔をあけて表示する

ここでは「もちてん＝」というメッセージと，現在の持ち点，つまり変数Mの内容を表示させたいので，以下のようにPRINT命令を使います．

```
130  PRINT "もちてん=";M
```

次に，（2）の点を賭ける部分です．これにはINPUTを使えばよいでしょう．

```
140  INPUT "かけてん=" ; K
```

変数Kは，賭け点を表しているわけです．

最後に，（3）の部分です．140行で入れた賭け点Kと，現在までの持ち点Mとを比べて，もしKがMより大きければ（1），つまり130行に戻ればよいわけです．何かの条件を判断し「もし〜ならば〜せよ」というときには，IF〜THEN〜という命令を使います．

IF〜THEN〜の使い方

もし　　　ならば
IF [条件] THEN [条件が成立したときに実行する命令]

- A=B（AとBが等しい）　　A<>B（AとBが等しくない）
- A>B（AがBより大きい）　　A>=B（AがB以上）
- A<B（AがBより小さい）　　A=<B（AがB以下）

★ A，Bはどんな変数，式でもよい
★ [条件]が成立していればTHENのあとの命令を実行し，成立していなければ，何もしないで次の行に移る

例）
- IF A=3 THEN PRINT A
  もしA=3ならAの内容を表示せよ
- IF A<>3 THEN 100 ………THENのあとのGOTOが省略されている
  もしAが3でなければ100行へ進め

これを使うと，プログラムは，

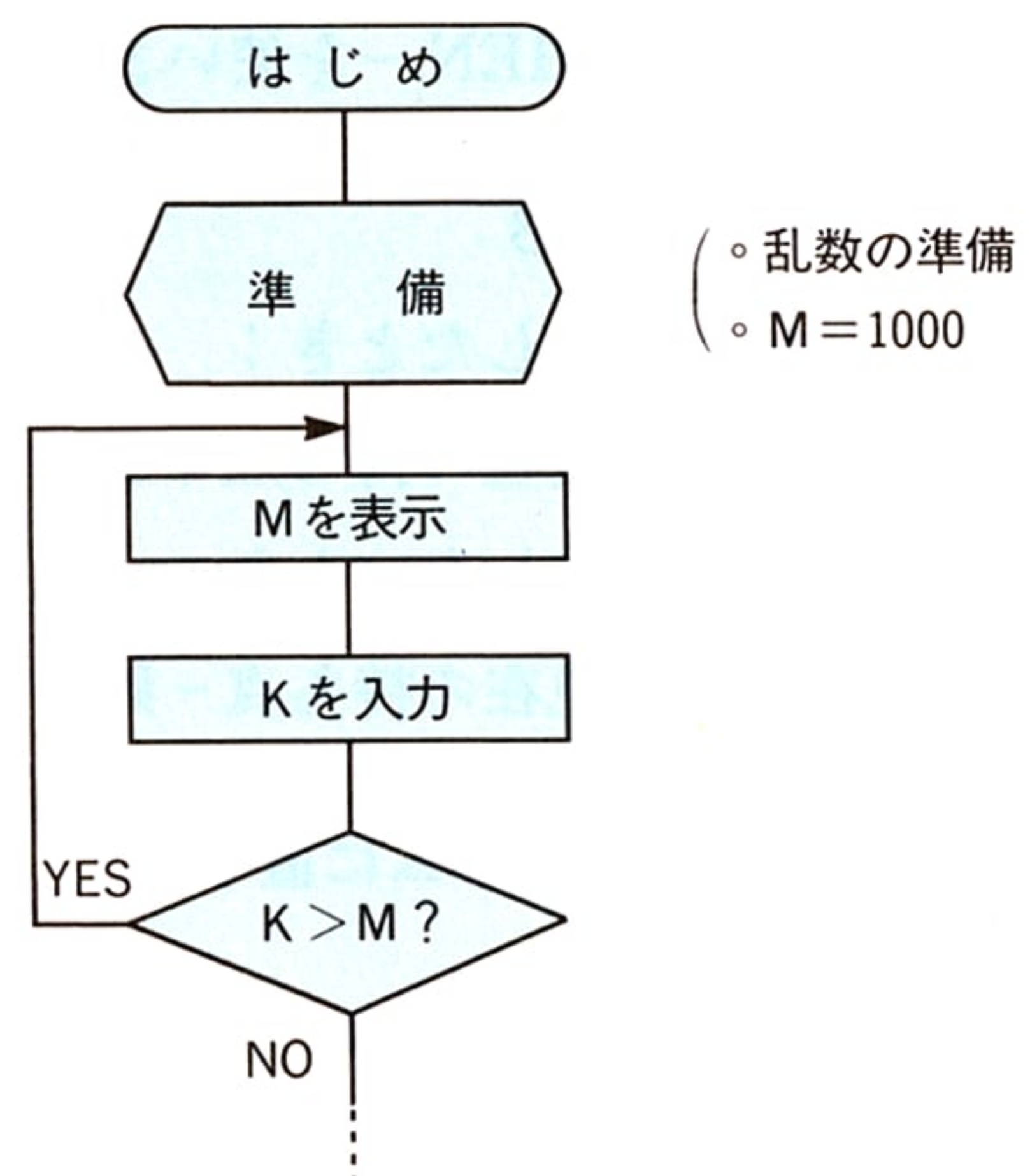
```
150  IF  K>M  THEN  130
```

となります．

THENのあとは，正確にはGOTO 130とすべきところですが，省略も可能なので，ここでは，省略しています．これでK>Mならば130行に戻り，そうでなければ次の行に進みます．

ここまでを流れ図（フローチャート）といわれる形に整理して，右にまとめておきます．

はじめ
準　備
Mを表示
Kを入力
K>M？
YES
NO

◦乱数の準備
◦M=1000

数字を表示する部分

賭け点が決まったら，乱数を使って０から９までの数字を2つ算出して，表示させます．2つの数字はA, B２つの変数で表します．

```
160  A=INT (RND ( 1 )*10)
170  B=INT (RND ( 1 )*10)
180  PRINT
190  PRINT  A ; B
200  PRINT
```

変数A, Bの内容を表示させているのは190行ですが，その上下にPRINTだけの行があります．PRINTは単独で用いると改行を行うので，表示を見やすくするために使用しています．

当たりはずれの計算

次に，乱数を使って算出した2つの数字を比較し，新しい持ち点を求めるプロセスを書かなければなりません．この部分を，さきほどちょっと載せた図の形で整理してみましょう．

ここでもIF～THEN～を使います．

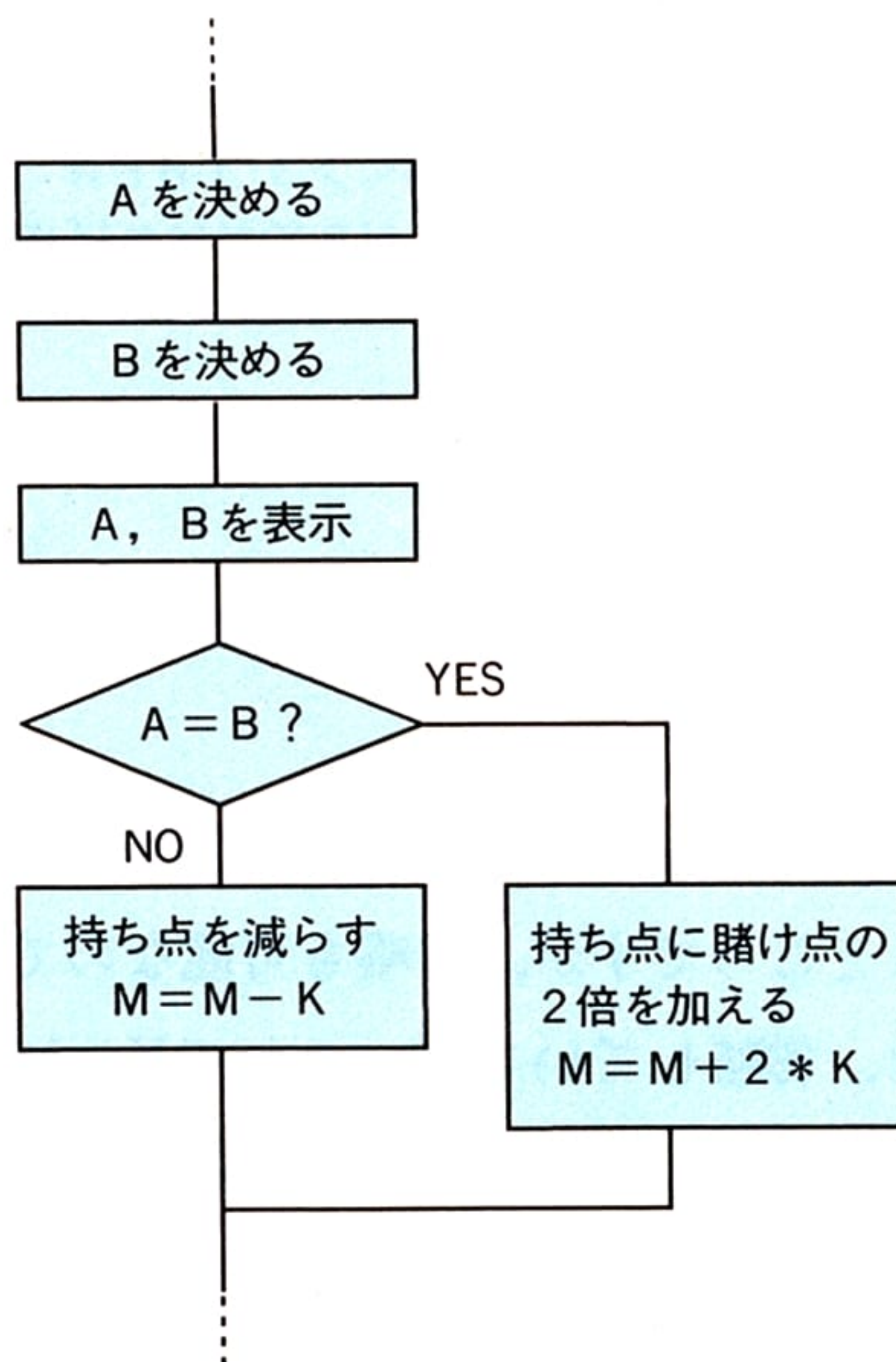
（１）条件：A＝B

（２）条件が成立したとき：
　　持ち点＝現在の持ち点＋賭け点の2倍

（３）条件が成立しないとき：
　　持ち点＝現在の持ち点－賭け点

A を決める
B を決める
A，B を表示
A＝B ?
YES
NO
持ち点を減らす
M＝M－K
持ち点に賭け点の
２倍を加える
M＝M＋２＊K

この判断をプログラムに直すと，

```
210  IF  A=B  THEN  M=M+2*K
220  M=M-K
```

としたくなります．が，現実にはこれではうまくいきません．A＝Bでないときは，何もせずに220行にいきますから持ち点を減らします．しかし，A＝Bのときにも，持ち点に賭け点の2倍を加えたあと，やはり220行に行ってしまうのです．

これを防ぐには，M＝M＋2＊Kのあと，220行をとばして次の行にいかなければなりません．そこで，210行を次のようにします．

```
210 IF A=B THEN M=M+2*K:GOTO 230
```

コロン［：］で区切って並べることにより，1つの行に2つ以上の命令を書くことができます．この例では，THEN以下がひとまとまりの部分として扱われるので，A＝Bのときは，持ち点の計算をしたあと，220行を超えて230行から実行します．

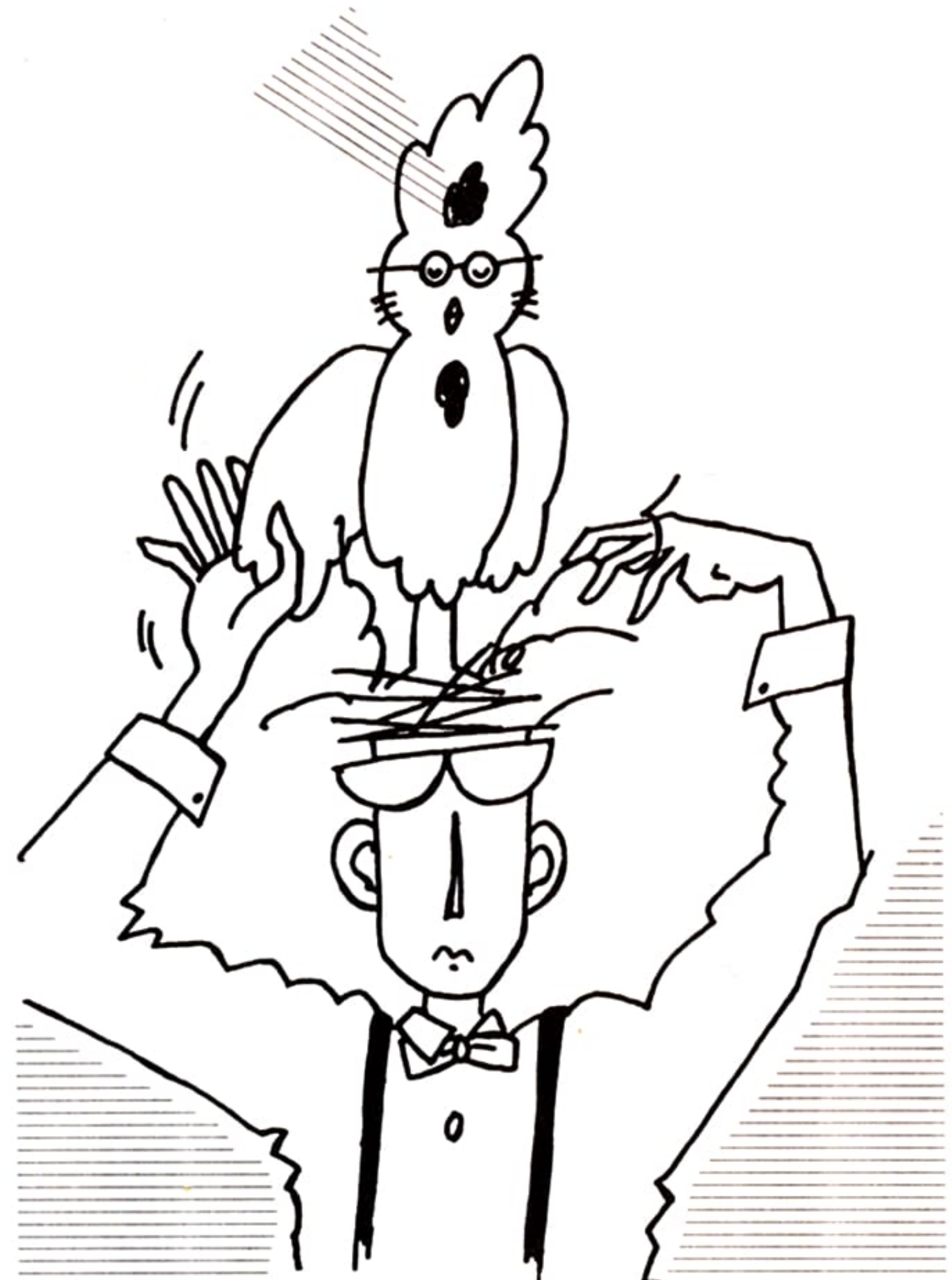
なお210行を実際に入力していくと，画面上の1行には収まりません．が，気にせずに入力を続けると，カーソルが自動的に次の行へ移動します．最後まで打ったら RETURN してください．

持ち点のチェック（終わりの判定）

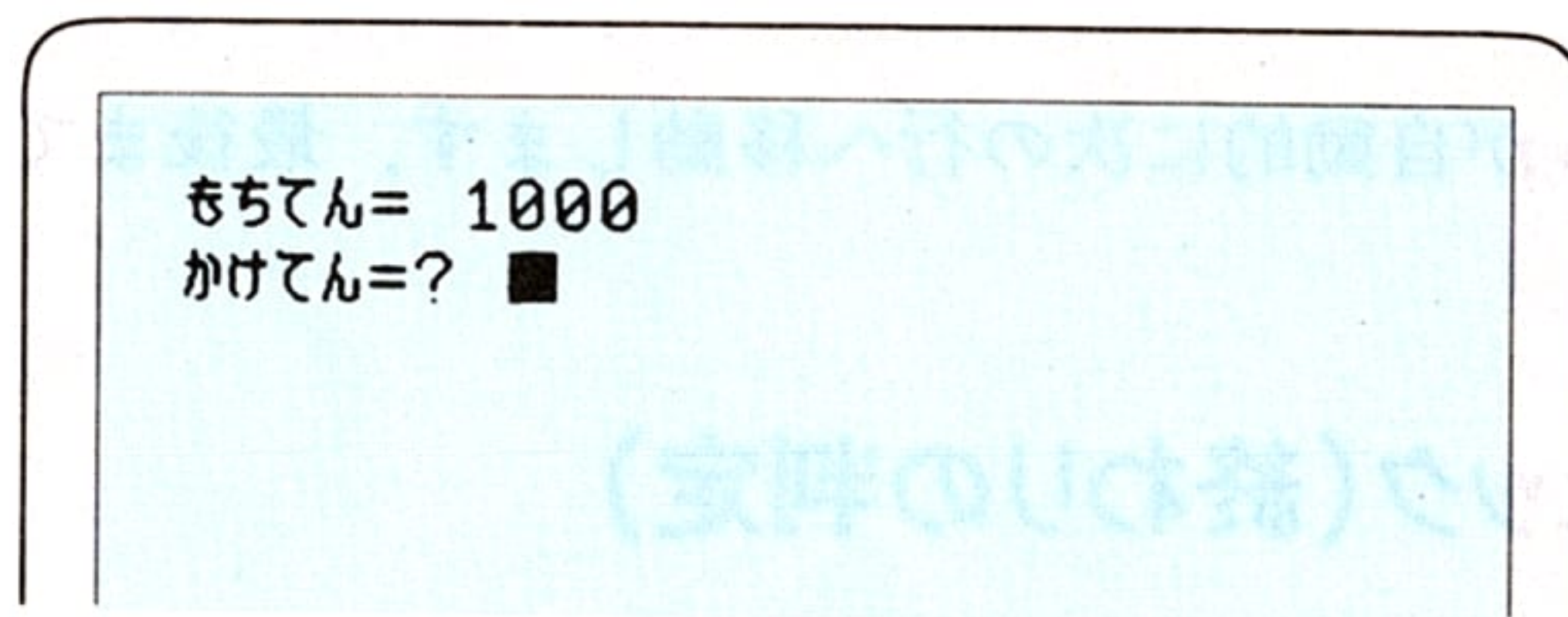
最後に持ち点のチェックをして，持ち点がゼロより大きければ続け，そうでなければ終わりにします．

```
230 IF M>0 THEN 130
240 PRINT "おしまい"
250 END
```

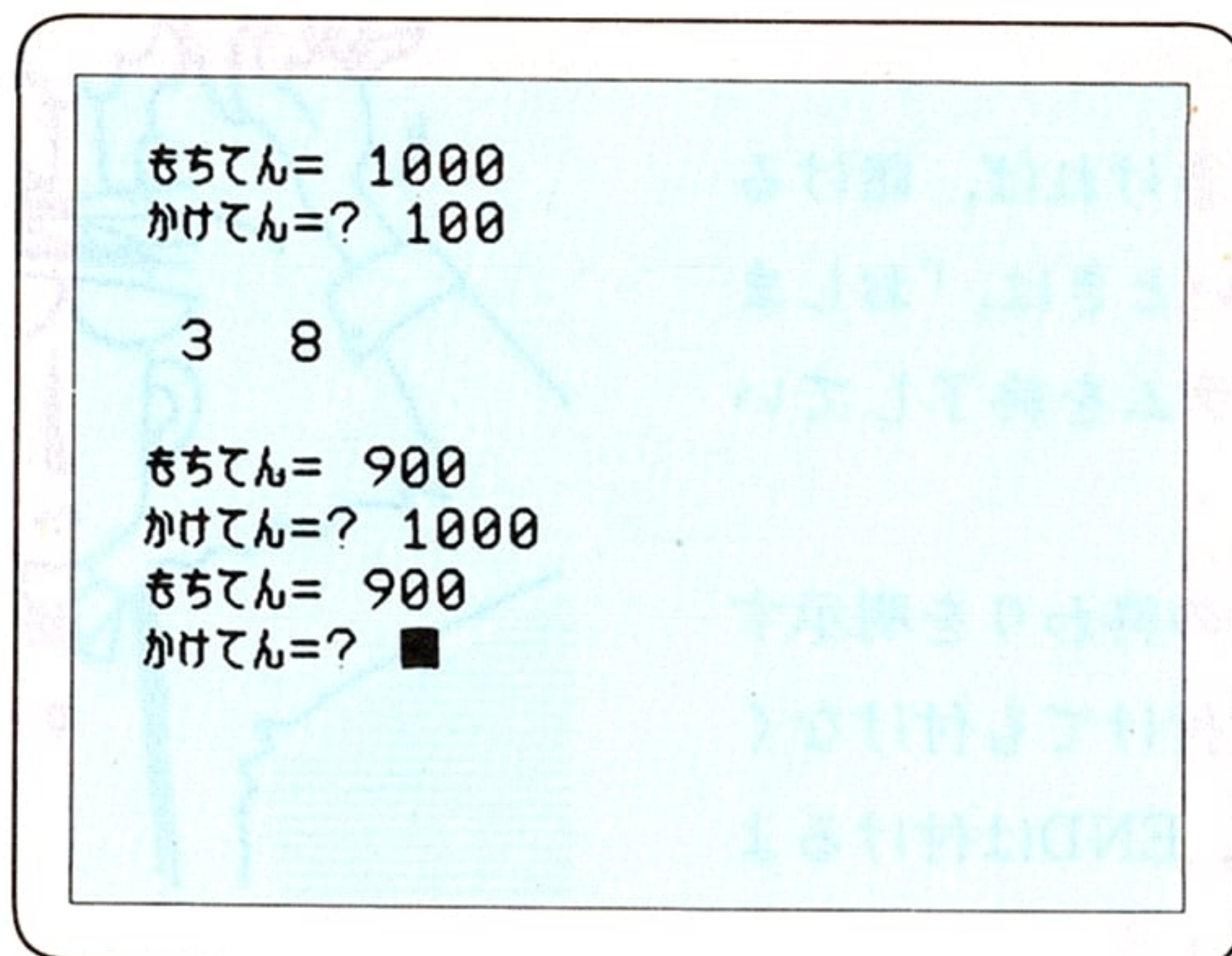
持ち点がゼロより大きければ，賭ける部分へ戻り，そうでないときは，「おしまい」と表示してプログラムを終了しているわけです．

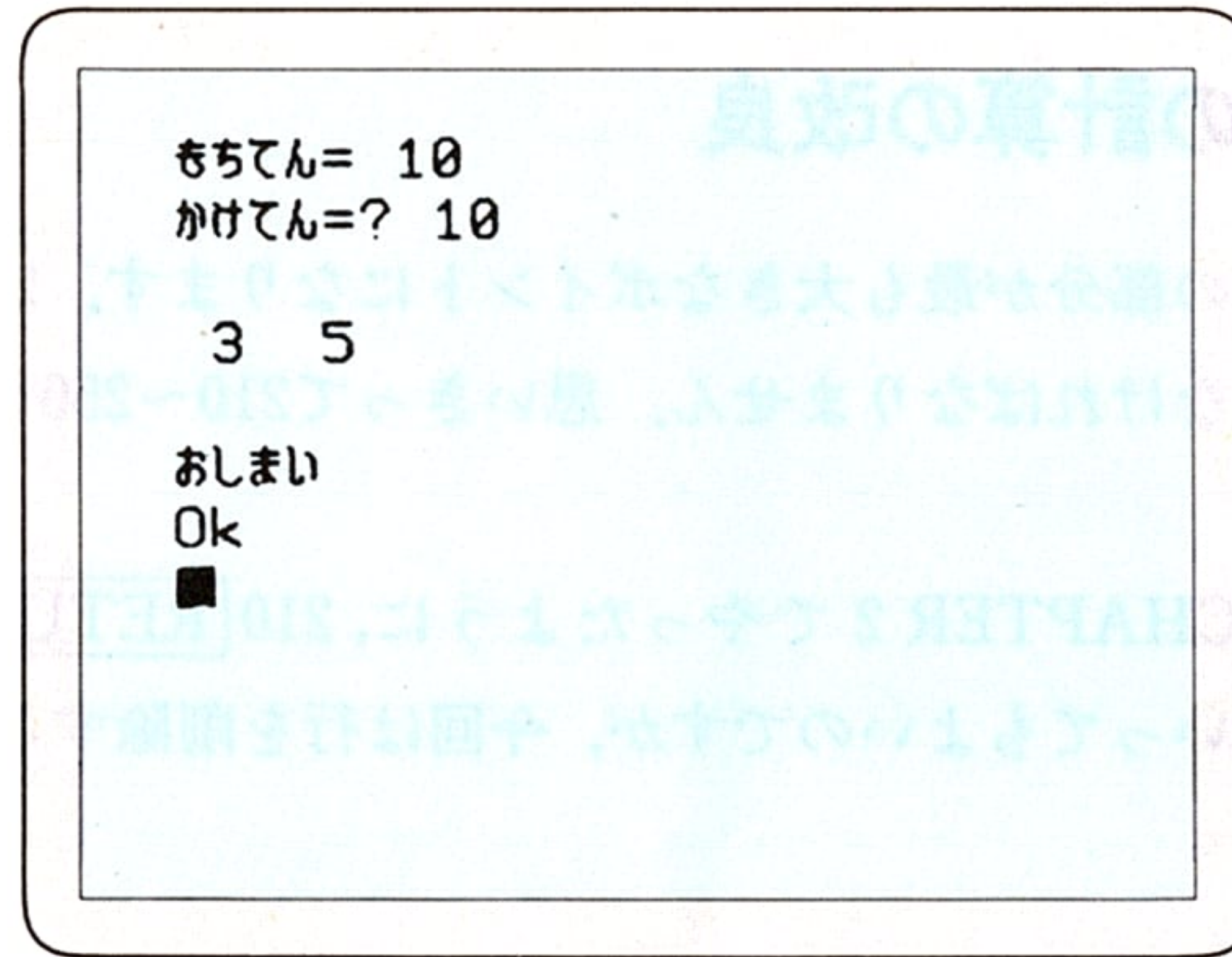
ENDは，プログラムの終わりを明示する命令で，ここでは，付けても付けなくてもかまいません．が，ENDは付けるようにした方がよいでしょう．

MSX BASIC 3-2 スロットマシンの改良

ではさっそく，3-1で入力したプログラムを試してみることにしましょう．プログラムを打ち込んでリストをとり，間違いがないことを確認したら実行します．

```
もちてん= 1000
かけてん=? ■
```

画面には，現在の持ち点が表示され，賭け点をどのくらいにするか聞いてきます．持ち点の範囲内で点を賭けると，2つの数字が表示されます．この2つの数字が一致していれば，賭け点は2倍になって戻ってきますが，一致していなければ没収されます．

持ち点が残っていると，再び点を賭け，持ち点がなくなるまで繰り返します．持ち点を超えて賭けようとすると，もう一度賭け点を入れるように求めてきます．

```
もちてん= 1000
かけてん=? 100

 3  8

もちてん= 900
かけてん=? 1000
もちてん= 900
かけてん=? ■
```

持ち点がなくなる(ゼロ以下になる)と，終わりになります．

```
もちてん= 10
かけてん=? 10

 3  5

おしまい
Ok
■
```

持ち点がある限り何回でも賭けられますが，途中であきてしまった人は，CTRLキーとSTOPキーを同時に押してください．プログラムは止まります．

とりあえず，意図していたとおりのスロットマシンはできました．しかし，2つ窓のスロットマシンなどあまり見かけませんね．そこで次の目標として，このプログラムを修正して3つの窓のスロットマシンを作りましょう．

なお，このプログラムは持ち点がなくなったときにしか終わりません．ある点以上になったらこちらの勝ちにして，気持ちよくプログラムが終わるようにもしましょう．

まず，窓を3つにすることによって起こる修正点をまとめます．

○ もう1つの数字を乱数から算出して表示する

○ 数字がもう1つ増えるので，賭け点の増え方を，3つとも同じなら3倍，2つ同じなら2倍に直す

では，プログラムを手直ししてみましょう．3-1で整理した5つのステップのうち，初めの2つはそのままでよいので，3ステップ目から説明します．

数字を表示する部分の改良

ここでは，もうひとつ数字を乱数で発生させて表示します．新しい変数の名前をCとすると，追加，修正する行は次の2つです．

```
175  C=INT (RND (1) *10)
190  PRINT  A ; B ; C
```

当たりはずれの計算の改良

今回の改良では，この部分が最も大きなポイントになります．210行から250行まで，かなり大幅な変更をしなければなりません．思いきって210～250行を消してしまうことにしましょう．

行の削除の方法は，CHAPTER 2 でやったように，210 RETURN ，220 RETURN ……と1行ずつ消していってもよいのですが，今回は行を削除する命令，DELETEを使ってみます．

DELETE　初めの行番号－終わりの行番号

この命令は上の型で使います．消したい部分の初めの行番号と，終わりの行番号を指定すると，その間の行をまとめて消します．ここでは，

DELETE　210－250 RETURN

とします．リストを取れば，210～250までの行が消えているのがわかるでしょう．

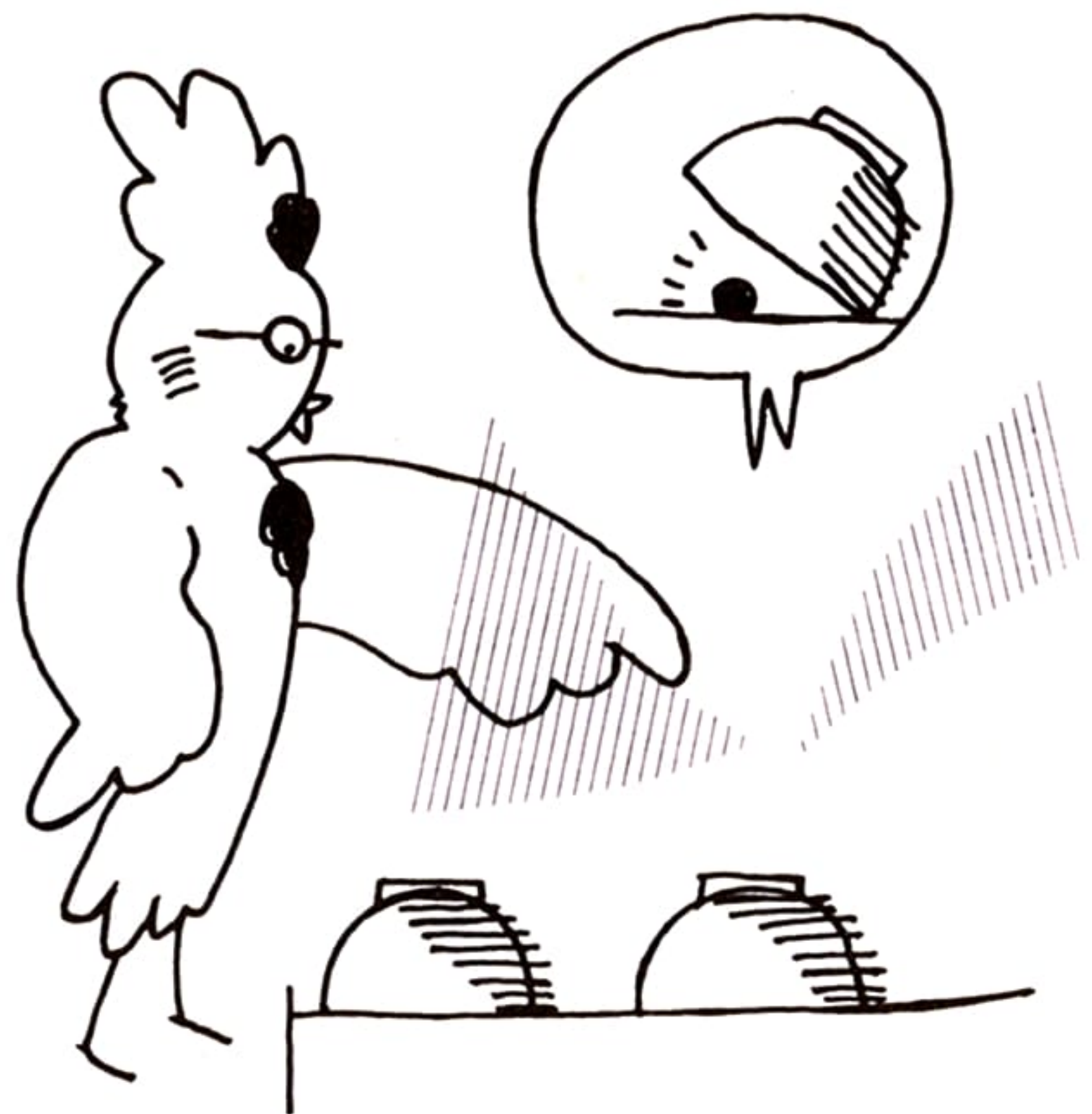
さて，当たりはずれの判定と賭け点の計算をしてみましょう．

まず，3つとも当たったときは，

```
IF A=B=C THEN M=M+3*K
```

としたいところですが，そうもいきません．残念ながらBASICでは同時に2つ以上の条件を判断するときに工夫がいるのです．

A=B=Cという条件を分解してみると，A=Bで，かつB=Cということになります．BASICでは「かつ」という条件をANDで表しますので，このプログラムは，

```
210  IF A=B AND B=C THEN M=M+3*K : GOTO 240
```

となります．これで，A=B=Cのときは，持ち点に賭け点の3倍を加え，次の部分(240行)に移るようになりました．

次に，2つの数字が一致するケースです．これを細かく整理すると，A=BまたはB=CまたはC=Aという，3つの条件のどれかが満たされている，ということになります．「または」を表すBASICの言葉はORです．これを使えば，2つの数字の一致を判定して賭け点を2倍にする部分は，

```
220  IF A=B OR A=C OR B=C THEN M=M+2 *K : GOTO 240
```

ということになります．

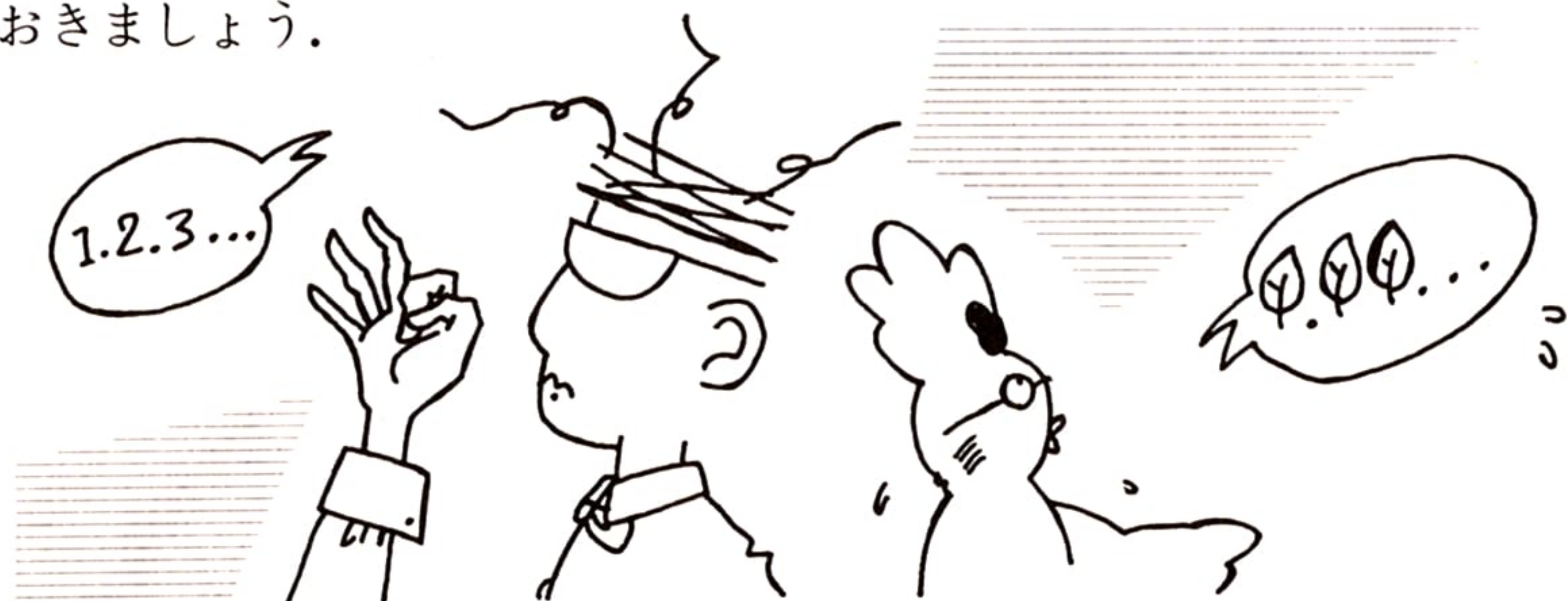
3つの数字がひとつもそろわないときには，点を没収するだけなので簡単ですね．

```
230  M=M-K
```

持ち点のチェックの改良

今回は，持ち点が1万点以上になったら，プログラムを終わらせることにします．1万点以上のときに，ただ終わりにするのもつまらないので，「こうさん！」と表示させることにしましょう．プログラムは次のようになります．

```
240  IF M<=0 THEN 270
250  IF M>=10000 THEN PRINT "こうさん！" : GOTO 270
260  GOTO 130
270  PRINT "おわり"
280  END
```

これでひとまずプログラムを完成させたことになります．全体のリストと実行画面をまとめて載せておきましょう．

リスト

```
100 'すろっとましん
110 X=RND(-TIME)
120 M=1000
130 PRINT"もちてん=";M
140 INPUT"かけてん=";K
150 IF K>M THEN 130
160 A=INT(RND(1)*10)
170 B=INT(RND(1)*10)
175 C=INT(RND(1)*10)
180 PRINT
190 PRINT A;B;C
200 PRINT
210 IF A=B AND B=C THEN M=M+3*K:GOTO 240
220 IF A=B OR A=C OR B=C THEN M=M+2*K:GOTO 240
230 M=M-K
240 IF M<=0 THEN 270
250 IF M>=10000 THEN PRINT"こうさん!":GOTO 270
260 GOTO 130
270 PRINT"おわり"
280 END
```

```
もちてん= 1000
かけてん=? 200

 6  8  3

もちてん= 800
かけてん=? ■
```

```
もちてん= 9900
かけてん=? 100

 6  6  5

こうさん!
おわり
Ok
■
```

修正のためのリスト表示

ところで，1回でうまくプログラムが動いた人はよいのですが，おそらくそのような人は少ないのではないでしょうか．コンピュータは，ほんのちょっとしたミスも許してくれません．ミスをしてエラーメッセージが出たとき，どう直せばよいでしょうか．

エラー直しの基本は，打ち込んだリストと，本に載っているリストをよく見比べることです．プログラムが大きくなると，リストが画面1枚には収まらなくなってしまうときがありますが，このようなときは一部分のリストだけを表示できれば便利です．このためのLIST命令の書き方をまとめておきます．

LISTの使い方

(1) ＬＩＳＴ　行番号
　特定の行のリストを画面に表示する　例）ＬＩＳＴ　１０

(2) ＬＩＳＴ　－行番号
　プログラムの最初から指定した行番号までのリストを表示する　例）ＬＩＳＴ　－１２０

(3) ＬＩＳＴ　行番号－
　指定した行番号からプログラムの終りまでのリストを表示する　例）ＬＩＳＴ　２６０－

(4) ＬＩＳＴ　行番号－行番号
　行番号で指定した範囲のリストを表示する　例）ＬＩＳＴ　１５０－４２０

(5) ＬＩＳＴ．
　エラーでプログラムがとまったときに，問題となっている行のリストを表示する

なお，LIST．はファンクションキーの9番に入っています．エラーメッセージが出たとき，[SHIFT]を押しながら[F・4]キーを押せば，間違えた行のリストが表示されます．

ミスが自分で見つけられて自分で直せるようになれば，あなたのプログラミングのセンスも，かなりアップしてきたといえるでしょう．

MSX BASIC

3-3 グルグルまわる数字

3－1，3－2とだんだんに改良してきて，スロットマシンゲームの原型ができあがりました．しかし，スロットマシンというには，なにかもうひとつ工夫が欲しいですね．

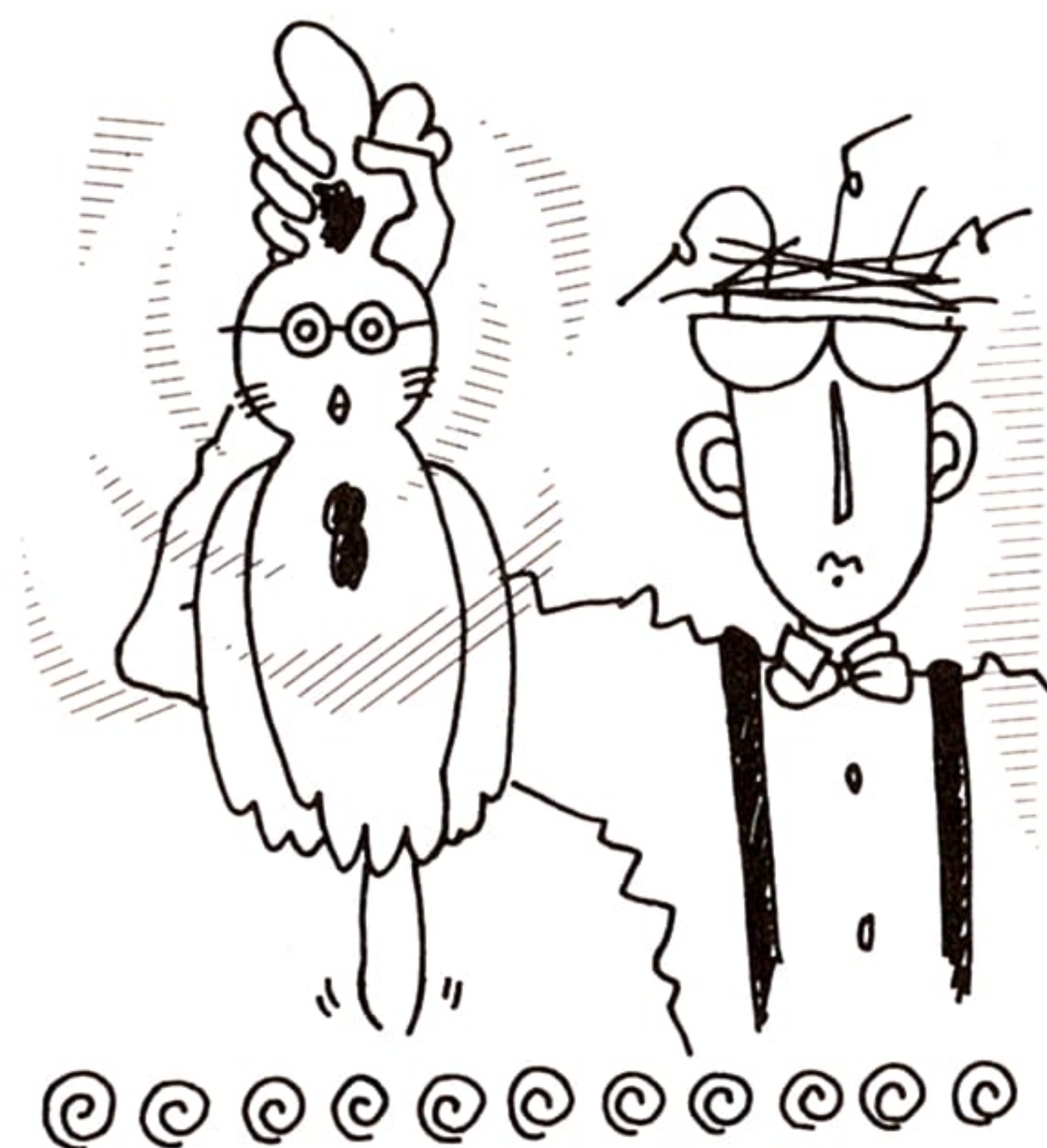
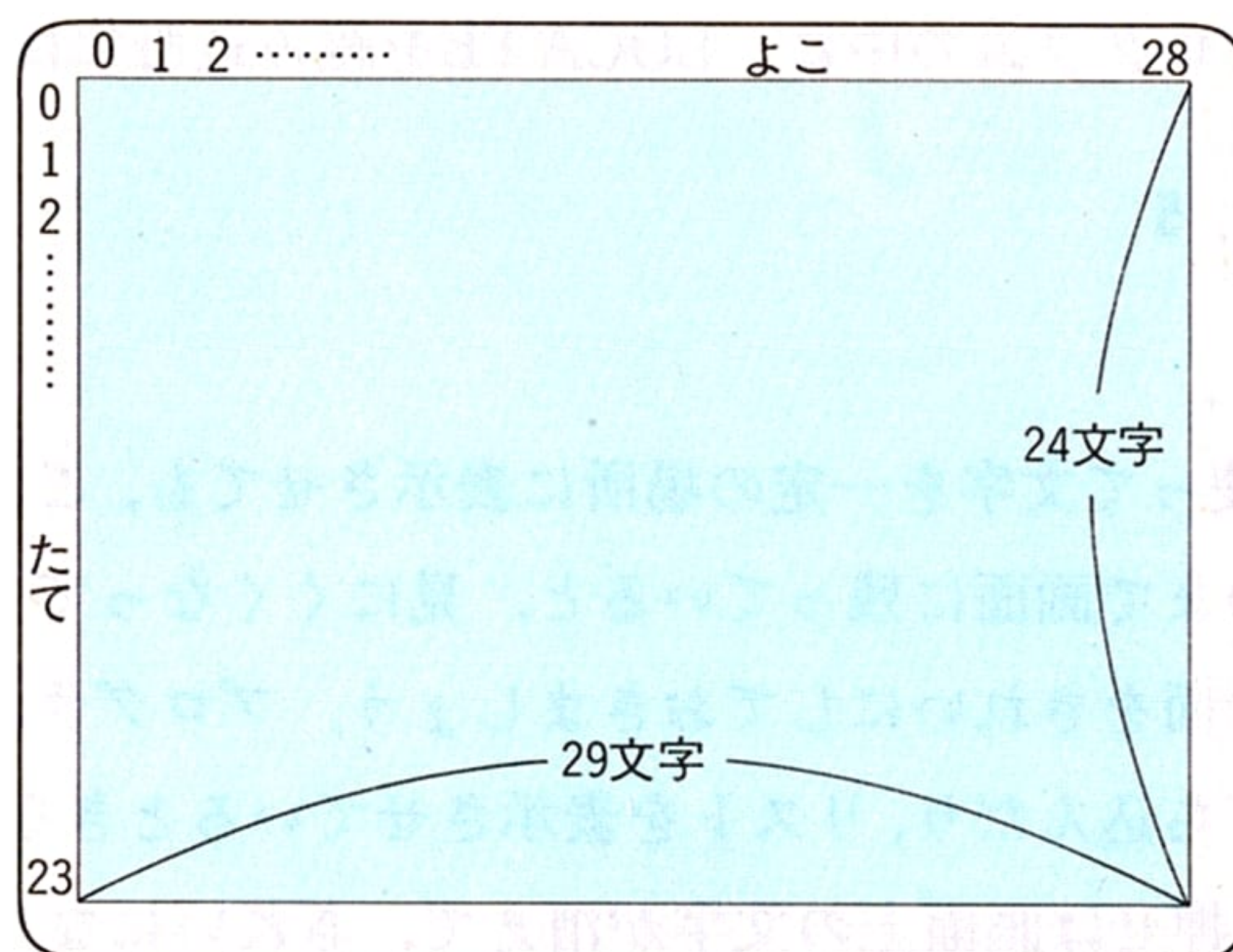
改良点の一番のポイントは，3つの数字の出かたです．本物のスロットマシンだったら，コインを入れたとたんに3つの絵がパッと出たりなどしません．ぐるぐると絵が回って，それを自分で止めるのです．

ここでは，スロットマシンの仕上げとして，数字をぐるぐる回して，もっと本物に近いスロットマシンゲームにしてみましょう．

ではいつものように，「数字をぐるぐる回す」という部分を具体的なレベルに落としていきます．その前に本物のスロットマシンをもう一度思い浮かべてください．3つの窓が並んでいて，それぞれの窓でいろいろな絵が，上から下へとグルグルグルです．

特定の位置に数字を表示する

まず，「窓」という言葉を具体化します．これは3つの数字の表示位置を，画面上の特定の位置に固定することで解決できるでしょう．フラフラと動き回る窓があるとしたら，これはもう窓ではありません．次に「上から下へとグルグルグル」ですが，これは次から次へと違った数字を表示させて解決します．

画面上の特定の位置に数字を表示する命令は，LOCATEです．この命令は，文字を表示する位置を，横，縦の番地で指定します．

文字を書く画面で位置を指定するための基準

MSXに電源を入れたときは，画面は横29文字，縦24文字の表示ができるようになっています．この，画面上の横，縦の位置をLOCATEを使って指定するわけです．LOCATE命令のひな形は次のとおりです．

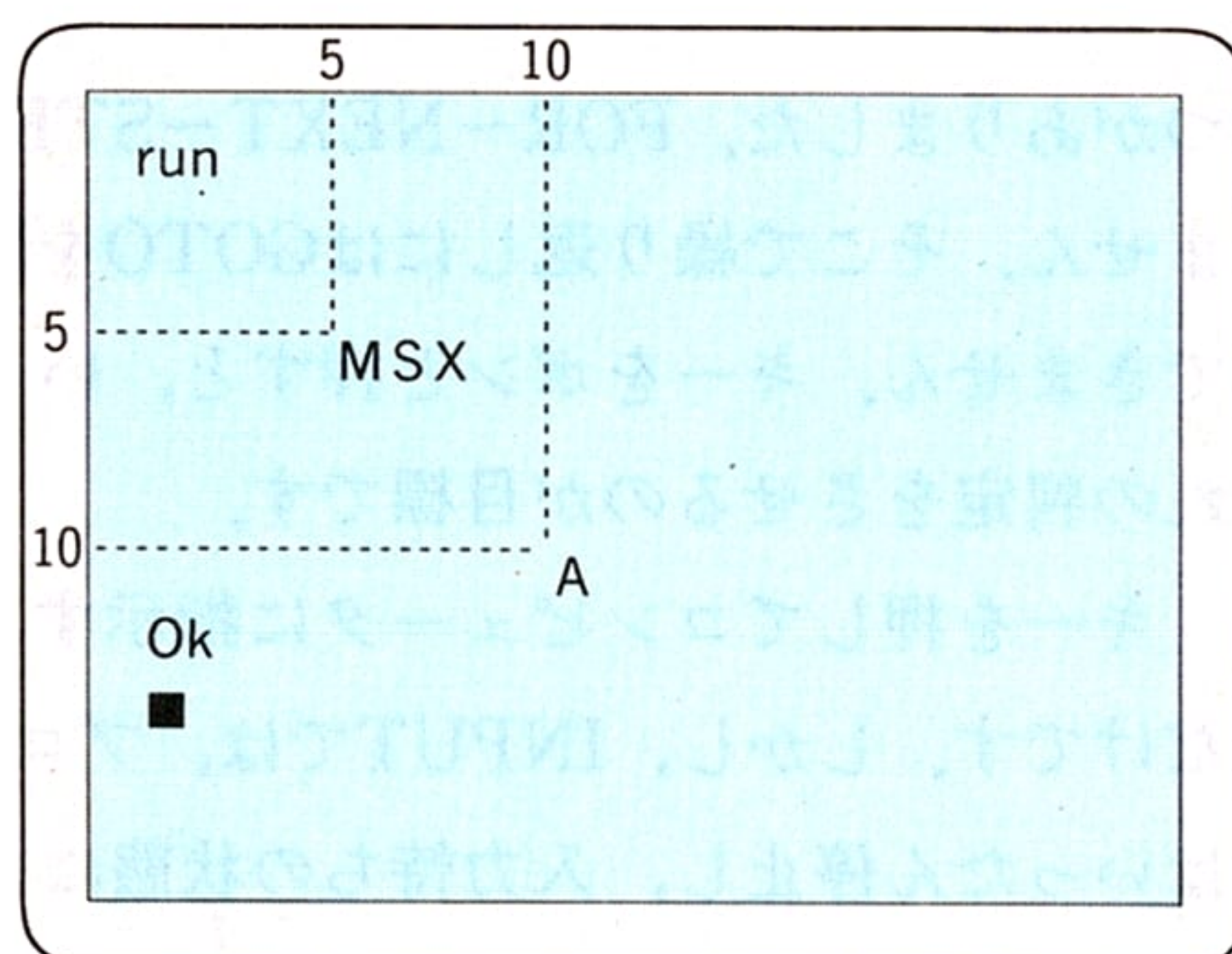
LOCATE　よこ，たて

よこ：0～28

たて：0～23(ファンクションキーの内容表示があるので普通23は指定しない)

例)　LOCATE　3，5

例として，次のようなプログラムを作ってみました．F・10を使うと画面を消してからプログラムを実行できます．これを使って実行しましょう．

```
list
10 LOCATE 5,5
20 PRINT"MSX"
30 LOCATE 10,10
40 PRINT"A"
Ok
■
```


SHIFT＋F・5(F・10)で画面を消してから実行させたところ

スロットマシンのプログラムの中に，LOCATEを組み込むには，

```
180  LOCATE 3 , 5
```

とすればよいでしょう．

さて，LOCATEを使って文字を一定の場所に表示させても，これまでのように現在の表示と関係のないものまで画面に残っていると，見にくくなってしまいます．そこで数字を表示する前に，画面をきれいにしておきましょう．プログラムが動いていないとき，つまりプログラムを打ち込んだり，リストを表示させているときなどは，SHIFTキーとHOMEキーを同時に押せば画面上の文字が消えて，きれいになります．

それと同じ働きをするBASICの命令として，CLSが用意されています．数字を表示させる前にCLSを入れて，画面をきれいにしておきましょう．3-2で作ったプログラムに，

```
155  CLS
```

を追加してください．

あとはLOCATEで決めた場所に，繰り返し数字を表示させればよいわけです．

好みのタイミングで数字を止める

では次に，3つの数字を繰り返し表示させ，自分の好みのタイミングで止める方法を考えてみましょう．

これまで覚えた繰り返しに使われる命令には，FOR～NEXT～STEPと，GOTOの2つがありました．FOR～NEXT～STEPは有限回しか繰り返せないので，ここでは使えません．そこで繰り返しにはGOTOを使います．が，これだけでは途中で止めることができません．キーをポンと押すと，いままで動いていた数字が止まって，当たり，はずれの判定をさせるのが目標です．

キーを押してコンピュータに指示する命令として，いままでに出てきたのはINPUTだけです．しかし，INPUTでは，プログラム中にINPUTが表れたところでプログラムはいったん停止し，入力待ちの状態になってしまいます．数字を次々と表示し続けながらキー入力を受け付けるためには，INPUT命令ではどうにも対応できません．

このスロットマシンのプログラムに限らず，何かゲームでも作ろうとするときは，即

時にキーに反応するような機能が欲しくなります．そうした機能をもった命令のひとつが，INKEY＄です．INKEY＄の中には，今押されているキーの文字が蓄えられているのです．Aのキーを押せばAという文字，Bのキーを押せばBという文字が，INKEY＄の中に蓄えられます．もしどのキーも押されていなければ，INKEY＄の中は何も入っていない状態になります．

IF～THENと，INKEY＄を使ってキーが押されているかどうかを判定する部分は，

```
200  IF INKEY$= "" THEN 160
```

のようになります．INKEY＄＝""というのが，何もキーが押されていない，ということをBASICで書き表したものです．これで，キーボードから何も字が入っていない状態であれば160行に戻って，再びA，B，Cの3つの数字を乱数で決め，それらの数字を横3，縦5の位置から表示することを繰り返します．

プログラム全体のリストを示すと次のとおりです．

リスト

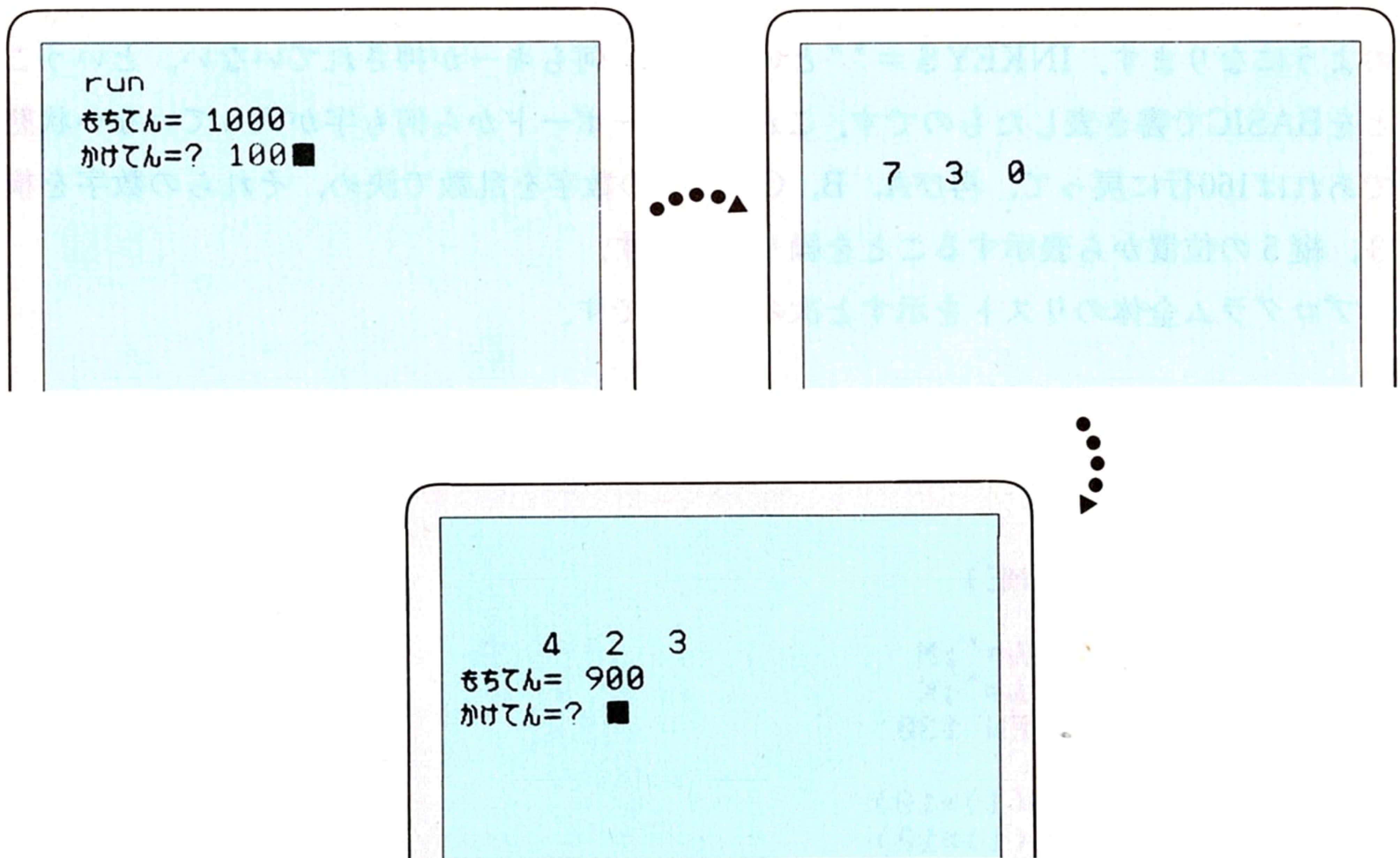
```
100 'するっとましん
110 X=RND(-TIME)
120 M=1000
130 PRINT"もちてん=";M
140 INPUT"かけてん=";K
150 IF K>M THEN 130
155 CLS
160 A=INT(RND(1)*10)
170 B=INT(RND(1)*10)
175 C=INT(RND(1)*10)
180 LOCATE 3,5
190 PRINT A;B;C
200 IF INKEY$="" THEN 160
210 IF A=B AND B=C THEN M=M+3*K:GOTO 240
220 IF A=B OR A=C OR B=C THEN M=M+2*K:GOTO 240
230 M=M-K
240 IF M<=0 THEN 270
250 IF M>=10000 THEN PRINT"こうさん!":GOTO 270
260 GOTO 130
270 PRINT"おわり"
280 END
```

さて，これでスロットマシンゲームの改良は，ひとまず終了です．改良に改良を重ねて作り上げたプログラムなのです．さっそくプログラムを実行して遊んでみましょう．プログラムを実行するときには F・5 のキーを使うと便利でしたね．

実行した直後は，改良前のプログラムの動きと変わりありません．でも，賭け点を入力すると一度画面が消されて，３つの数字がぐるぐる回り始めます．数字を止めるには，何か適当なキーを押します．キーを押すタイミングで数字の並びが違ってくるので，ずっと本物のスロットマシンらしくなってきました．

```
run
もちてん= 1000
かけてん=? 100■
```

```
7  3  0
```

```
4  2  3
もちてん= 900
かけてん=? ■
```

動きの部分の改良なので，読んでいるだけではわかりにくいでしょうが，ほんの２，３行変えただけでゲームとしてのおもしろさは格段に違ってきます．もっと欲をだせば，色を変えたり，音を付けたりしたいところです．さらに贅沢をいえば，ただの数字がグルグル回るのではなく，きちんとデザインされた絵が動くようにしたいものです．

こういった工夫は，また４－４で行うとして，せっかく作ったプログラムですから保存しておくことにしましょう．テープにプログラムを保存する方法はもうわかりますね．忘れてしまった人は２－６をもう一度読み直してください．これであなたの２本目のプログラムが保存されたことになります．

COLUMN

「言語」にちょっと触れる

コンピュータの世界に関わりはじめると，「言語」ということばを耳にするようになるでしょう．この本を読みながらＭＳＸを扱っている限り，この「言語」という概念は意識しなくてもかまいませんが，それ以上の興味を持ち続けるあなたのために，「言語」にちょっと触れます．

コンピュータの世界で意味する「言語」とは，コンピュータに命令するときの話し方の取り決めのことです．それほど深い意味はありません．たとえばあなたのＭＳＸは，「ＭＳＸ　ＢＡＳＩＣ」という言語の取り決めどおりに，命令を実行する機械ということなのです．

では，コンピュータの言語は誰が考え出したものでしょうか．人間の使う言語が長い歴史を経て自然に発生していったのに対し，コンピュータの言語は，人間が人工的に作り上げたものなのです．コンピュータの言語を考え出すのは，一個人の場合もあれば，大学の研究室の場合もあり，また企業の開発スタッフのこともあります．

ＭＳＸ　ＢＡＳＩＣ の場合は，米国のマイクロソフトという会社が，マイクロソフトＢＡＳＩＣという言語をもとに，いくつかの改良を加えて作りあげたのです．そしてさらにルーツをたどっていけば，ＢＡＳＩＣという言語は1960年代の半ばに，ダートマス大学のケメニーとカーツという数学者によって作られたことがわかります．

当時ＢＡＳＩＣは，「初心者のための万能記号命令コード」と名付けられ，コンピュータの専門家ではない人々が機械に慣れ親しむため，という意図を持った言語だったのです．この基本構想は，20年以上たった今でも変りなく活かされているようです．

MSX BASIC

3-4 グラフに挑戦(1)

—合計・平均—

CHAPTER 2 ではさまざまな円を描き，CHAPTER 3 のここまでのsectionではスロットマシンゲームを作ってきました．コンピュータが初めて登場した1940年代には，まさか私たちのような普通の人々が，コンピュータでカラフルな円を描いてみたり，ゲームを作って遊ぶなどとは想像もつかなかったことでしょう．

さて，次なる課題はMSXにおける実用性の追求です．ここからの 3 つのsectionでは，MSXを使って計算をし，結果をグラフに表すことを目標にしましょう．

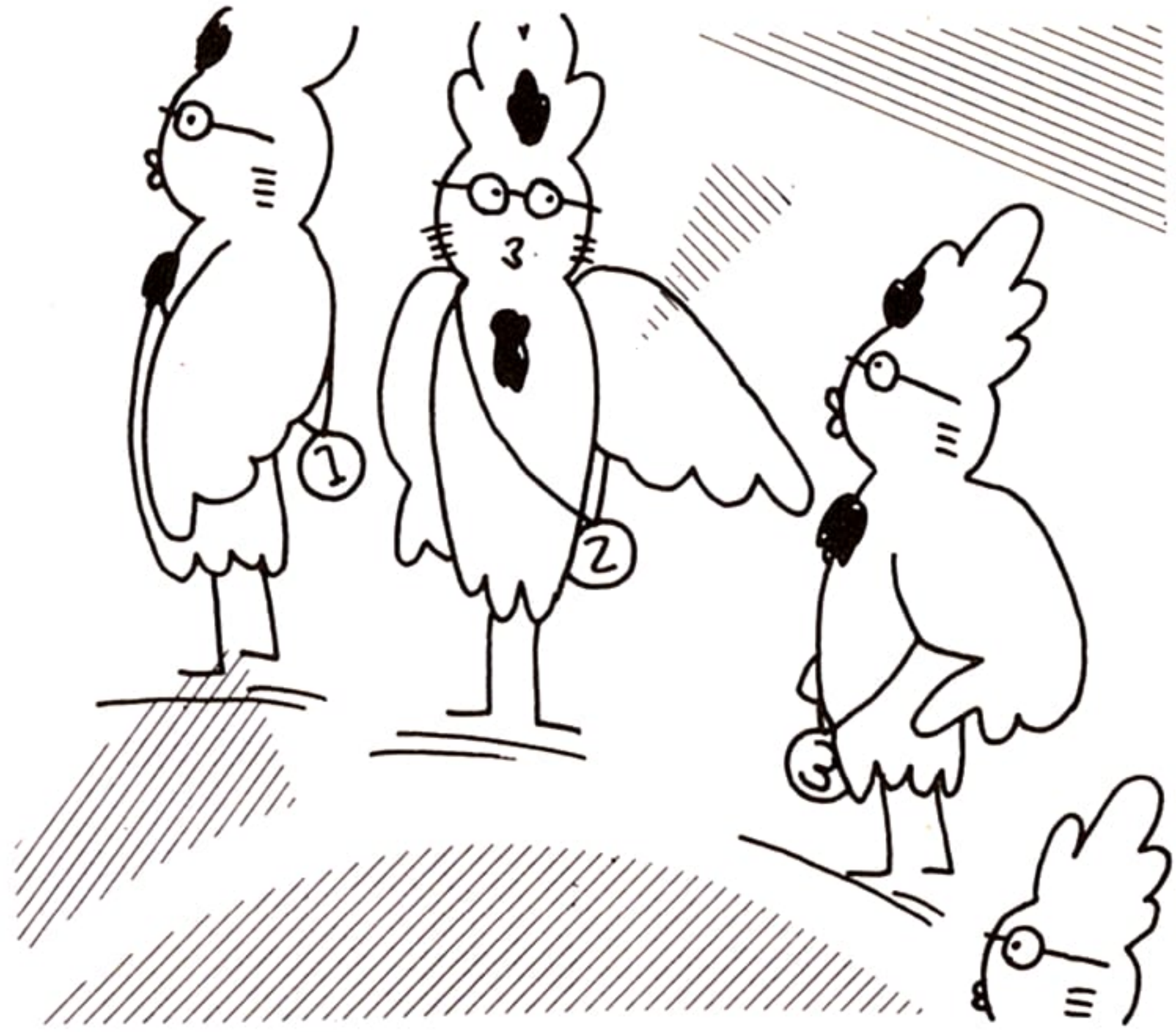
実用性追求のための練習問題

仲間で作っている野球チーム〝MSX〟の今シーズンの上位 5 人のホームランの数は右のとおり．
この 5 人のホームランの数の合計と平均を求めよ．また，各人の本数をグラフにして比較せよ．

名　前	ホームランの数
原	18
田　淵	16
中　畑	9
石　毛	6
江　川	4

実用性の追求にしては簡単すぎますが，それでもこれを100％実現するのは，骨の折れることです．手はじめに，この 5 人の合計と平均を計算させてみることにしましょう．

合計を計算するということを，例によって整理してみます．「合計」などといっても結局たし算の繰り返しであることはすぐにわかります．ここでは，ある変数をひとつ決め(Sとしておきます)，その変数に一人ひとりのホームランの数を加えていけばよいのです．この手順をさらに整理すると，次のようになります．

① 計算の対象となる人数を決める
② ①で決めた人数について
(1)各人のホームランの本数をキーボードから入力する
(2)入力した数を変数に足し込んでいく
という処理を繰り返す
③ 平均計算をする
④ 合計と平均を表示する
⑤ 終わり

プログラムの作成

それでは，この手順をBASICのプログラムにしてみましょう．プログラムの先頭には，3-1で説明したアポストロフィ[']を使って，プログラムの内容をコメントとして残しておきます．

```
100 'ごうけい と へいきん
```

まず，①の計算の対象となる人数を決める部分です．人数を表す変数はNとしておきます．いろいろな場合にも対応できるように，INPUTを使ってキーボードから計算の対象となる人数（Nの値）を入力できるようにしておくとよいでしょう．

```
110 INPUT "にんずう=";N
```

次に②の部分です．ここでは，(1)各人のホームランの本数を，キーボードから入力する，(2)入力されたデータを変数に加える，という2つの作業を①で決めた人数分だけ繰り返します．この部分をプログラムにすると，次のようになります．

```
120 FOR I=1 TO N
130  INPUT D
140  S=S+D
150 NEXT I
```

120行のFORと150行のNEXTで，繰り返す範囲と回数を決めています．120行のFORでは，STEPが省略されています．このようなときには，STEPの値は１とみなされ，Iの値は１，２，３，４…Nと変化します．

130行では各人のホームランの数をキーボードから読み込んでいます．このために使っている変数はDです．

140行のSが合計を出すための変数です．この行の働きを言葉で表現すると，「SにDを足して，新たなSとせよ」となります．BASICでは＝記号が使われるわけですが，意味としては，

```
S←S+D
```

のように矢印で表した方がよいものなのです．

これで②の部分はOKです．残りの部分も難しくありません．

```
160  AV=S/N
```

平均＝合計÷データの数ですから，このようになります．合計，平均の値を表示するところでは，もちろんPRINTを使います．

```
170  PRINT "ごうけい=";S
180  PRINT "へいきん=";AV
```

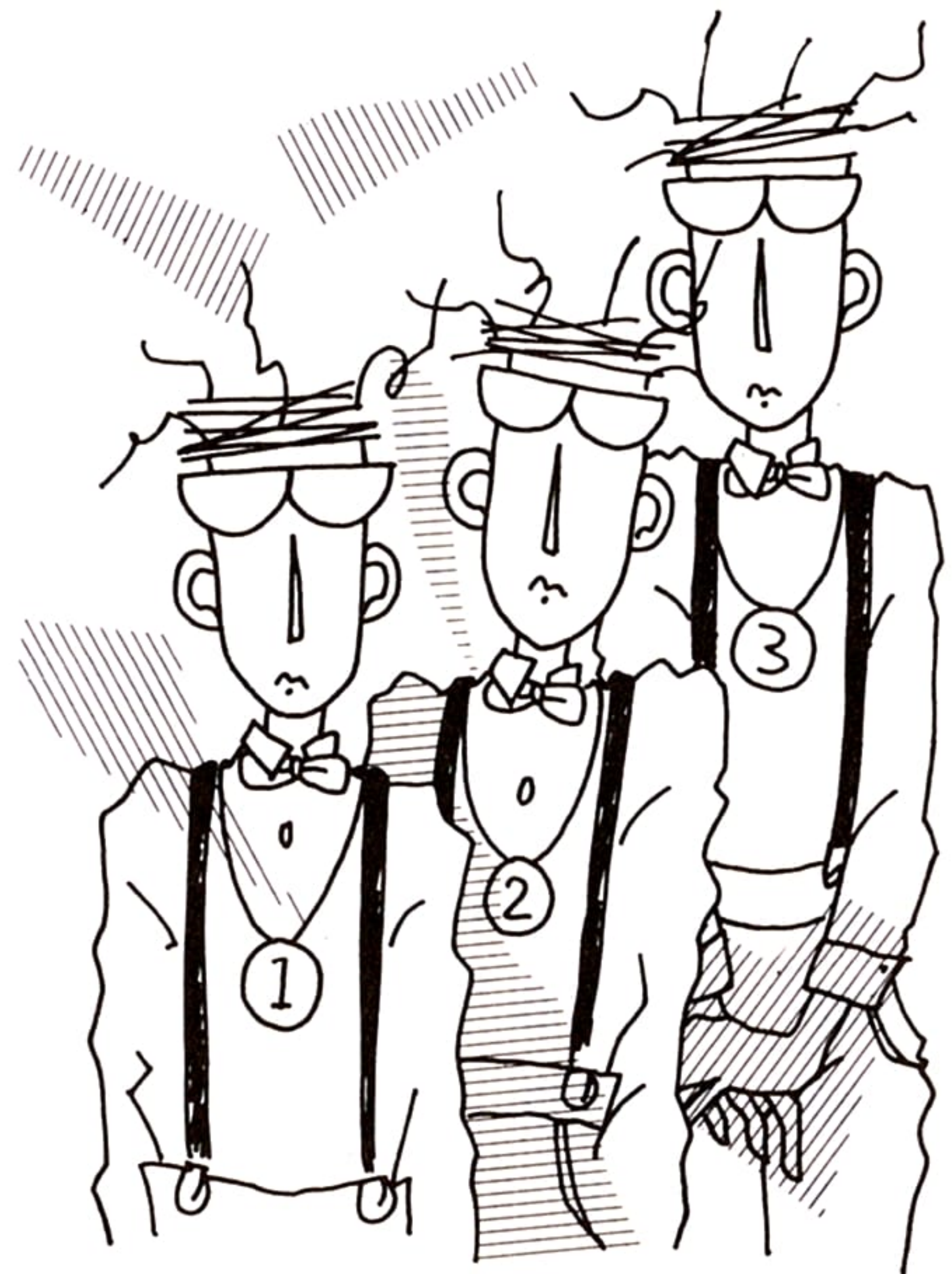
⑤の終わりの部分はプログラムの最後に，

```
190  END
```

を付け加えるだけです．

意外と簡単にできてしまいましたね．プログラミングの基礎知識をきちんと身に付けたうえで，問題を上手に整理すれば，数式の出てくるプログラムでも，それほど苦労せずに作れるのです．

ここで全体のリストを載せておくことにしましょう．

リスト

```
100 'ご"うけい と へいきん
110 INPUT"にんず"う=";N
120 FOR I=1 TO N
130  INPUT D
140  S=S+D
150 NEXT I
160 AV=S/N
170 PRINT"ご"うけい=";S
180 PRINT"へいきん=";AV
190 END
```

計算式については，もう何度か出てきていますが，四則の記号についてひととおりまとめておきます．

種　類	普通の記号	BASICの記号
たし算	＋	＋
ひき算	−	−
かけ算	×	*
わり算	÷	／

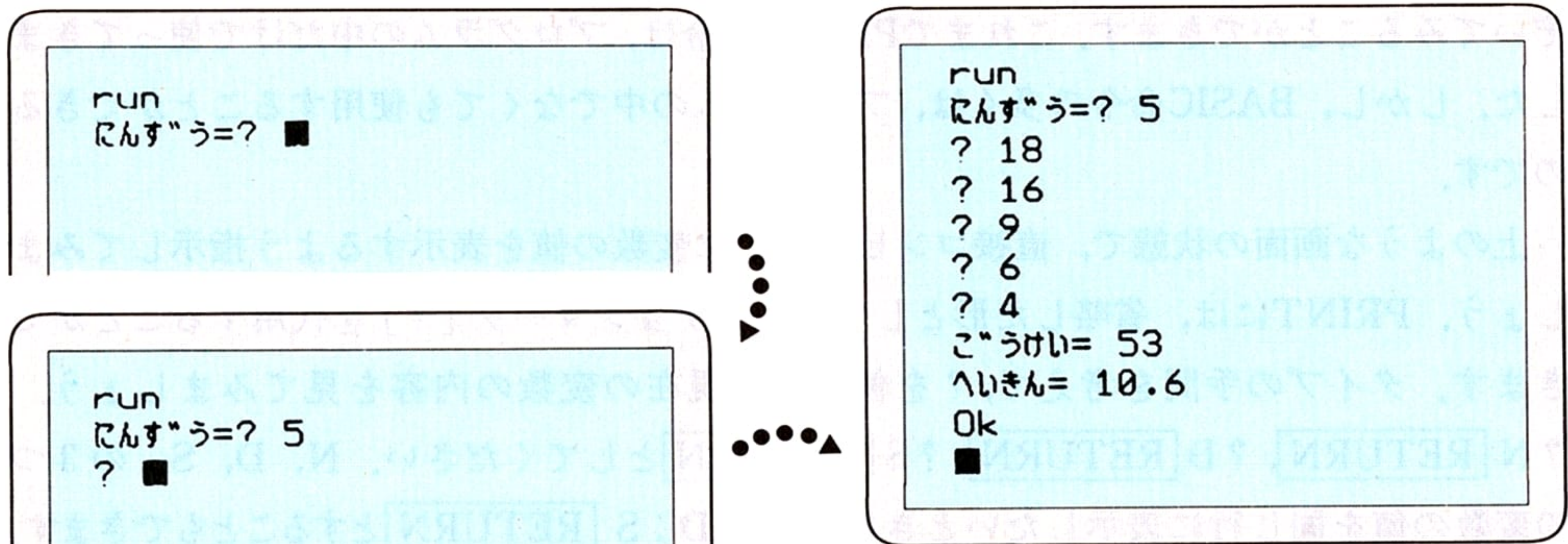
四則演算の記号

MSXのBASICでは，上に挙げた四則演算(たし算，ひき算，わり算，かけ算)以外にもべき乗や三角関数，LOGなどを扱うことができます．

プログラムが理解できたらさっそく実行してみましょう．

```
run
にんず"う=? ■
```

```
run
にんず"う=? 5
? ■
```

```
run
にんず"う=? 5
? 18
? 16
? 9
? 6
? 4
ご"うけい= 53
へいきん= 10.6
Ok
■
```

まず人数を聞いてくるので，5 RETURN とします．人数を入れたあとは，5人分のホームランの数を入れていきます．ここでは18 RETURN，16 RETURN，9 RETURN，6 RETURN，4 RETURN と入力しました．答えは合計53，平均10.6となります．結果が信用できないという人は，電卓で検算してみてください．

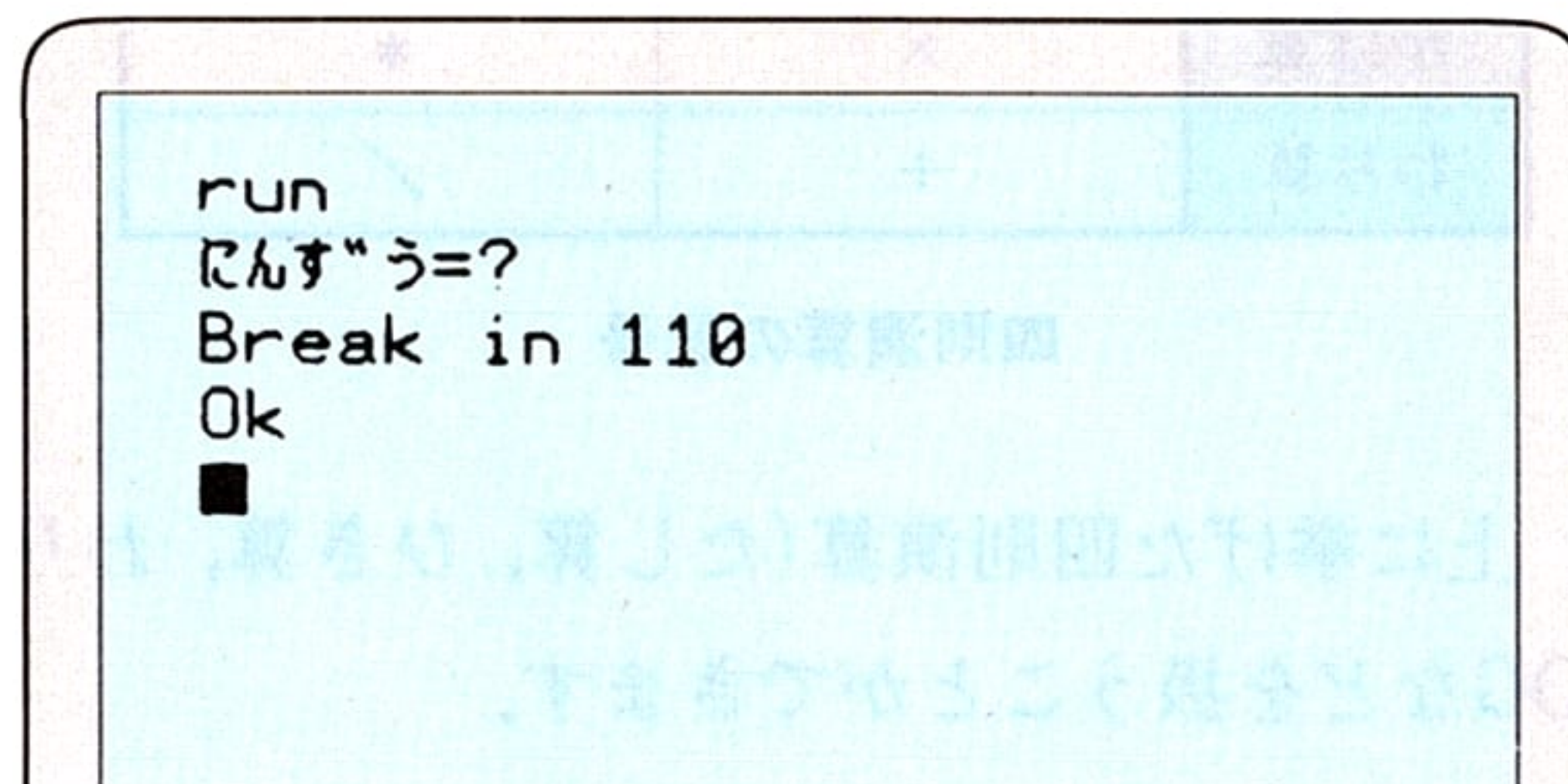
プログラムの中断とチェック

プログラムを実行させて結果を確認するだけでなく，プログラムの途中途中で，いろいろな変数がどのような値になっているかを見ておきましょう．この作業は，プログラムの流れが間違っているかどうかを調べるのにもよい方法です．

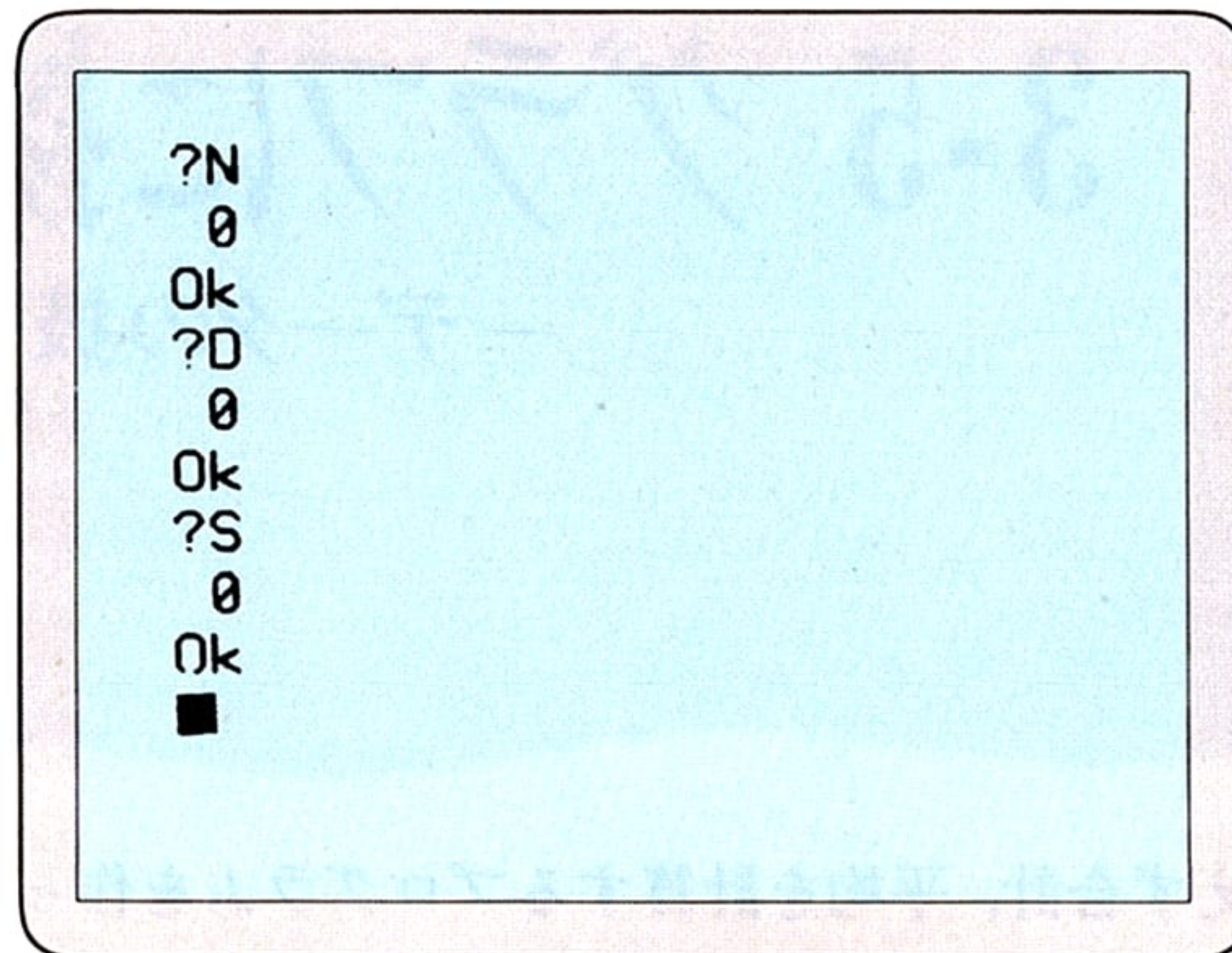
プログラムを再び実行して画面に，

にんずう＝?

と表示された段階で CTRL キーと STOP キーを同時に押してプログラムをいったん止めてください．

```
run
にんずう=?
Break in 110
Ok
■
```

画面は上のようになり，プログラムは一時中断されます．このときに，変数の値をのぞいてみることができます．これまでPRINT命令は，プログラムの中だけで使ってきました．しかし，BASIC命令の多くは，プログラムの中でなくても使用することができるのです．

上のような画面の状態で，直接コンピュータに変数の値を表示するよう指示してみましょう．PRINTには，省略した形としてクエスチョンマーク［?］を代用することができます．タイプの手間を考えて，?を使って，現在の変数の内容を見てみましょう．?N RETURN，?D RETURN，?S RETURN としてください．N，D，S，の3つの変数の値を同じ行に表示したいときは，?N；D；S RETURN とすることもできます．

```
?N
 0
Ok
?D
 0
Ok
?S
 0
Ok
■
```

さて，このようにして中断したプログラムですが，必要な変数の中味を見たあとは再開させましょう．いったん止めたプログラムの実行を再開する命令はCONTです．CONTはファンクションキーの8番に登録されているので，SHIFTを押しながらF・3のキーを押すとよいでしょう．なおこのプログラムの変数は，次のように変化します．

行番号	I	D	S
120	1	0	0
130	1	0	0
140	1	18	18
150	2	18	18
120	2	18	18
130	2	18	18
140	2	16	34
⋮	⋮	⋮	⋮
130	5	6	49
140	5	4	53

← I＝2のとき，つまり2人目のデータを入れるとき，プログラムを停止し，?I ?D ?Sで変数の中味を見ると，I＝2，D＝18，S＝18となっている

プログラム実行中の変数の内容

あなたも実際にプログラムをいったん止めて，変数の中をのぞいてください．どのようなプロセスを経て計算が行われていくのかがわかってくると思います．

MSX BASIC

3-5 グラフに挑戦（2）

―データの扱い―

3－4では，とりあえず合計，平均を計算するプログラムを作ってみました．意外と簡単だったので気がゆるんでいる人もいることでしょう．しかし忘れないでください．私たちの目標は，グラフを完成させることです．

合計と平均ができたので，すぐにグラフを描けるかというとそうもいきません．グラフを描くためには，合計する前の個人のデータがきちんと蓄えられていることが必要だからです．3－4のプログラムを途中で止めて変数の中味をチェックすればわかるように，個人の成績は，変数Dに入っています．このプログラムのままでは，ホームランの本数は次のDの値が入力されると消えて，新しく入力されたホームランの本数に置き換わってしまいます．これではグラフを描くことはできません．なんとか，一つひとつのデータを取っておく工夫が必要になります．

これまでの知識の中から考えてみて，すぐに思い浮かぶ解決策は，人数分の変数を用意するという方法でしょう．

今までの知識で考えられる解決策

```
100 'ごうけいとへいきん
110 INPUT A  ┐
120 INPUT B  │ データの数だけ変数を用意する
130 INPUT C  │
140 INPUT D  ┘
      ⋮
200 S=A+B+C+D+………………
```

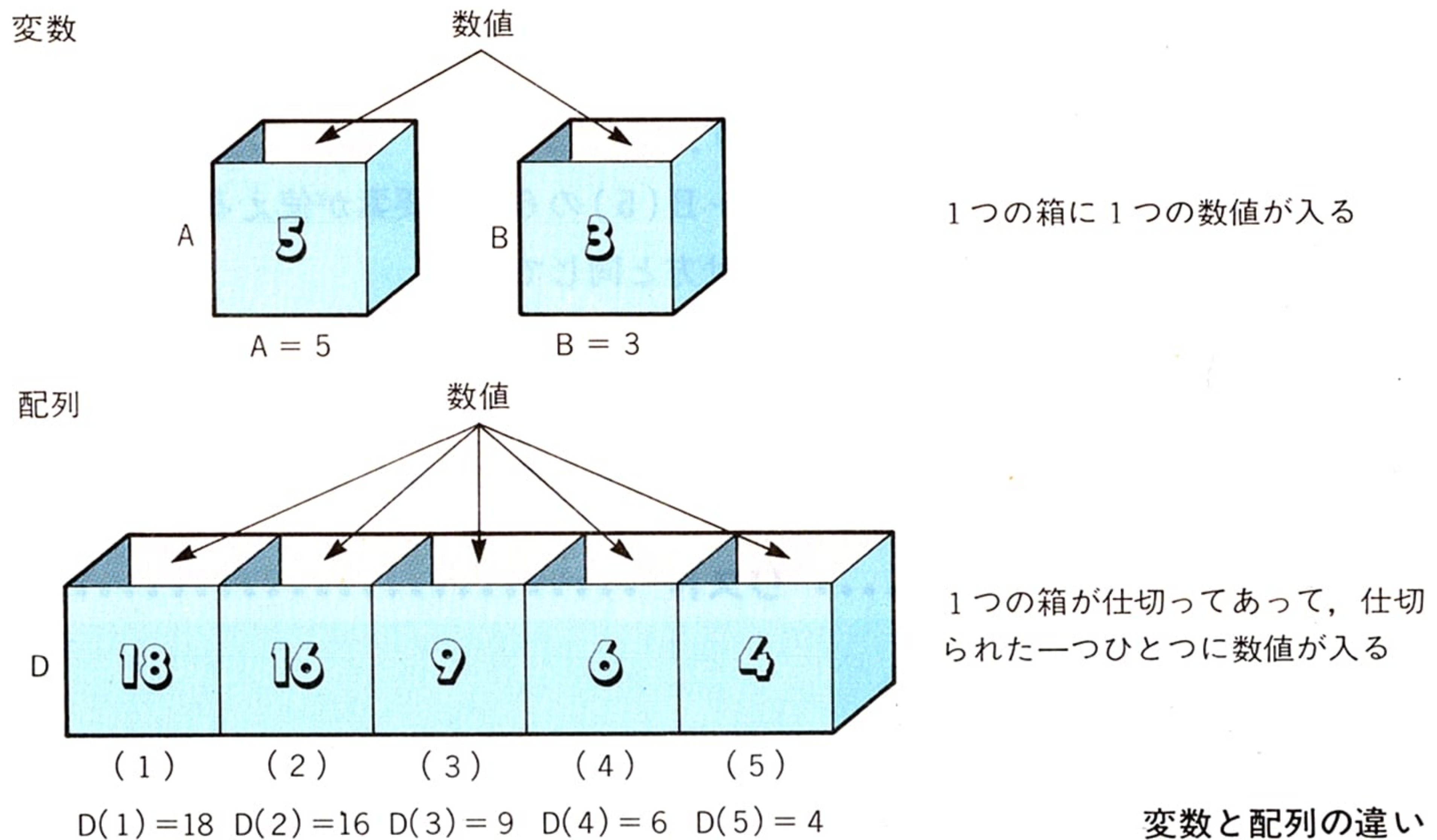
このようにすれば，確かに入力したデータの一つひとつは保存されます．しかし，これは非現実的で役に立ちそうもないことが，すぐにわかります．その理由は，変数の数に限りがある，入力できる数が固定される，プログラムが長く複雑になる，といったことです．

配列の利用

これらの問題点を解決するには，データを蓄えるのに，配列という新しい変数の形を使います．配列は同じ性質の多数のデータをまとめて扱うときに便利な変数です．これまでの変数と，配列変数の違いを図にしてみると次のようになります．

変数
数値
A 5
B 3
A＝5　B＝3
1つの箱に1つの数値が入る

配列
数値
D 18 16 9 6 4
(1) (2) (3) (4) (5)
D(1)＝18　D(2)＝16　D(3)＝9　D(4)＝6　D(5)＝4
1つの箱が仕切ってあって，仕切られた一つひとつに数値が入る

変数と配列の違い

変数が，一つひとつ独立した箱の中に数値を入れる，いわば一軒家のようなものだとすれば，配列はアパートのようなものだと思えばよいでしょう．上の図で，A，Bなどを変数の名前（変数名）と呼んだのと同じように，上のDのような配列の名前を，配列名と呼んでいます．配列名はいくつものデータを代表したグループの名前です．Dの中味は，上の図で1，2，3，4，5と書いてあるそれぞれの箱に入っています．配列の中にある数値を蓄える箱を，配列の要素と呼んでいます．使い方はこれから説明していくなかでしっかり覚えるようにしてください．

普通の変数に数を入れたりするときには，A＝3，B＝2のように変数名をそのまま使いました．配列はD（1）＝3，D（5）＝2のように，

配列名（添字）

という型で使います．配列の中で，何番目の要素であるかを表す1，2，3のような数字を添字と呼んでいますが，配列の要素は，上のように配列名と添字で指定することになるのです．

配列は他の変数と違って，使う前にこれから使用するということを宣言してやる必要があります．

DIM　配列名（添字の上限）

配列の使用宣言のひな形は上のとおりです．

たとえば，DIM　B（5）とするとB（0）～B（5）の6つの要素が使えるようになります．配列名の付け方の規則は，変数名の付け方と同じです．

プログラムの作成

では，実際にプログラムの中で配列を使ってみることにしましょう．

リスト

```
100 'ごうけい と へいきん
110 INPUT"にんずう=";N
115 DIM D(N)
120 FOR I=1 TO N
130  INPUT D(I)
140  S=S+D(I)
150 NEXT I
160 AV=S/N
170 PRINT"ごうけい=";S
180 PRINT"へいきん=";AV
```

まず，配列を使う宣言をしなければなりません．新たに付け加えた115行によってこの宣言を行っています．DIMで配列の使用宣言をするときに，添字の上限を具体的な5，10といった数字だけでなく，変数でも指定できることを知っておくと便利でしょう．

次に，一人ひとりのホームランの数を，D（I）に入れています（130行）．このように添字を変数で指定しておくと，Iの値を変えるだけでD（I）という配列のすべての要素を表すことができるので，便利です．

配列の要素を，D（1），D（2），D（3）のように具体的な数字を使って指定するのなら，普通の変数と手間は変わりません．D（I）という型を使い，要素の指定はIを変えることによって行うので，プログラムも短くて済むのです．

120～150行で，個々のデータ（この例ではホームランの本数）が配列Dの各要素に蓄えられることになります．具体的には，1人目のデータがD（1）に，2人目のデータがD（2）に，3人目のデータがD（3）に蓄えられます．3-4の例のとおり入力すれば，D（1）～（5）の中味は，

D(1)=18，D(2)=16，D(3)=9，D(4)=6，D(5)=4

となるはずです．

次に，入力されたデータが確かに保存されているかを調べるために，190行～240行を，下のリストのようにしてみました．

リスト

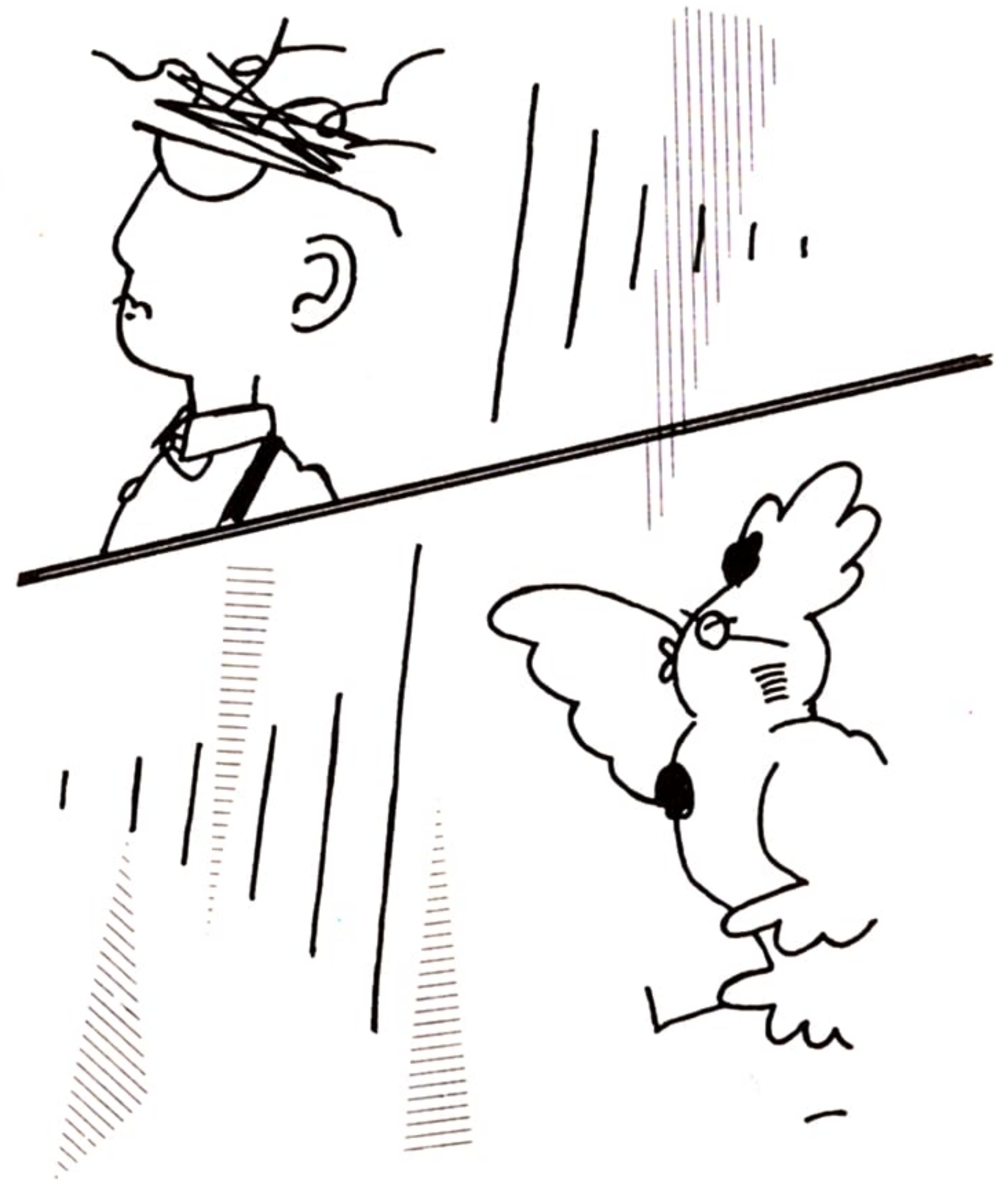
```
190 PRINT
200 PRINT"** せいせき **"
210 FOR I=1 TO N
220  PRINT D(I);
230 NEXT I
240 END
```

配列の中味の表示にも，FOR～NEXTを使ってＩを１からＮ（この例では５）まで，１つずつ変化させて，Ｄ（Ｉ）を表示させています．

210～230行をFOR～NEXTを使わないで書くと次のようになります．

```
210  PRINT D(1);
220  PRINT D(2);
230  PRINT D(3);
240  PRINT D(4);
250  PRINT D(5);
```

FOR～NEXTと配列をうまく組み合わせて使うと，プログラムがスッキリするのがよくわかりますね．

プログラムをひととおり打ち込んでみたら，実行して確認してください．

```
run
にんずう=? 5
? 18
? 16
? 9
? 6
? 4
ごうけい= 53
へいきん= 10.6

** せいせき **
 18  16  9  6  4
Ok
■
```

実行しているところを見れば，このプログラムのどの部分を実行しているのかは，わかりますね．にんずう＝？と表示されたら110行，合計・平均が表示されれば，これは170行，180行の結果，とわかるでしょう．

実行の追跡

このような短いプログラムではあまり心配ありませんが，少し込み入ったプログラムになると，今どこを実行しているのかわからなくなるときがあります．このようなときに威力を発揮する命令が，TRON，TROFFです．

プログラムを実行する前にTRON RETURN と打ち込んでから，実行してみてください．

```
tron
Ok
run
[100][110]にんずう=? 5
[115][120][130]? 18
[140][150][130]? 16
[140][150][130]? 9
[140][150][130]? 6
[140][150][130]? 4
[140][150][160][170]ごうけい= 53

[180]へいきん= 10.6
  :
```

TRONでプログラムを実行したところ
（190行以下の実行は省略している）

[行番号]の形で，プログラムがどこの行を実行中なのかを，画面に表示します．画面の表示は崩れてしまいますが，TRONを使うと，このようにプログラムの流れが一目瞭然です．頭の中が混乱してわけがわからなくなったら，TRONを使ってプログラムの流れを追ってみるとよいでしょう．

TROFFは，逆にこの行番号を表示する状態を解除するものです．

3-6では，いよいよグラフを描かせてみることにしましょう．

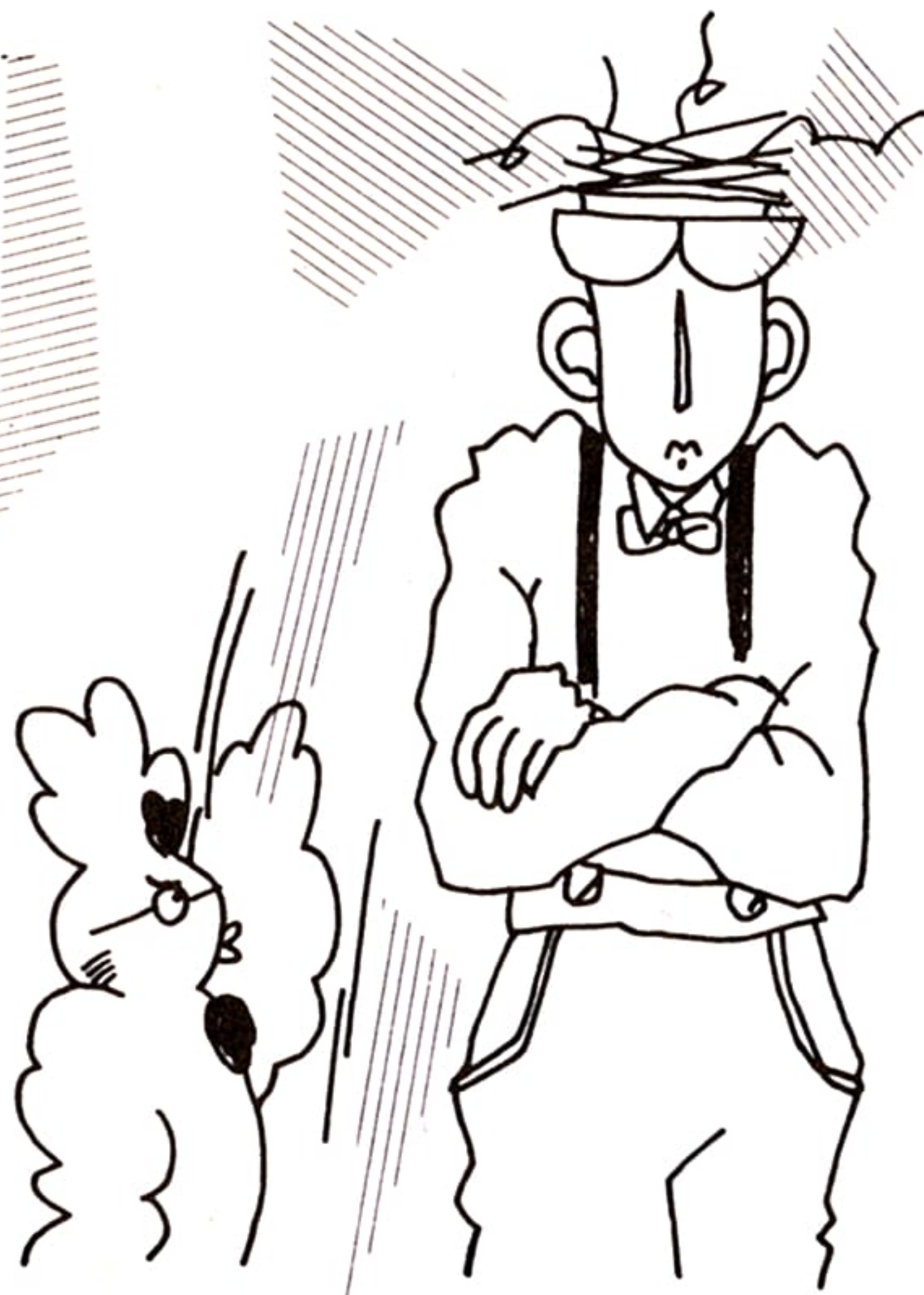
MSX BASIC 3-6 ついに棒グラフ完成

いよいよグラフを描くsectionにたどり着きました．今までの所は小さな処理単位にゆっくりと説明してきたので，ひととおり理解できたことでしょう．そこでグラフを描く前に，少し視点を変えてプログラム一般の働きを見直してみます．

コンピュータに仕事をさせるとき，プログラムの流れは大きく，入力→処理→出力の3つに分けることができます．

コンピュータが処理するもとになるデータを与えるのが入力の部分です．コンピュータにデータを入力する方法にはいくつかありますが，MSXの場合は，キーボード，ジョイスティック（ゲームのときなど），カセットテープなどが主に使われます．

データが与えられたら，次にそれを目的に応じて処理します．この処理は，たとえば合計，平均といった四則の計算で済むものもあるでしょうし，三角関数などを使う高度な計算であるかもしれません．他にも，データを大きい順や小さい順に並び替えたり，データを探し出す，といったことも処理に含まれます．

データを入力し，処理しても，それを人間の目に見えるようにしなくては意味がありません．処理された結果を，文章や表の形，グラフの形など，さまざまな表現を使って人間の目に見えるようにする部分が，出力の部分なのです．

グラフの選択

このように整理すると，グラフを描く作業は，入力→処理→出力の3つの作業のうちの，出力の部分を改良するのだということがわかります．そこで，これまでに作ったプログラムのうち，入力と処理の部分は，そっくりそのまま使うことにしましょう．

リスト

```
100 'ご"うけい と へいきん
110 INPUT"にんず"う=";N        人数の入力
115 DIM D(N)
120 FOR I=1 TO N
130  INPUT D(I)               入力と合計の計算   処理(加工)
140  S=S+D(I)
150 NEXT I
160 AV=S/N ........平均の計算
```

出力の部分は，合計・平均の表示と，データの表示を行っていたわけですが，この部分を改良して，データはグラフの形で表示させ，合計，平均はこれまでのように数字で表示させるようにします．そのためには，出力の部分を大修正する必要があるので，思い切ってこの部分を消してしまいましょう．

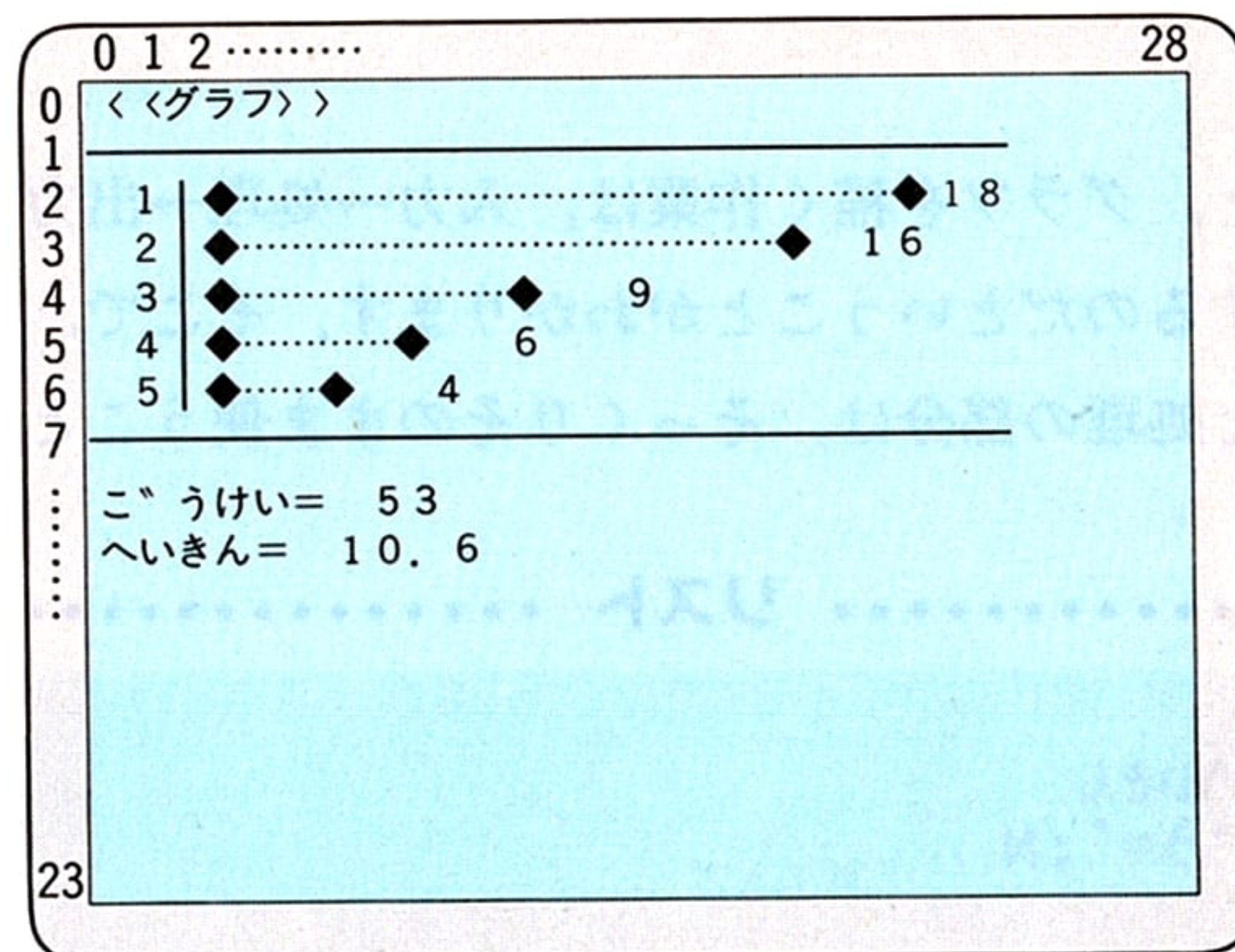
DELETE 170－240 RETURN

これで準備はOKです．

ところで，グラフといってもたくさんの種類があります．棒グラフ，折れ線グラフ，円グラフなど，目的に応じてさまざまなものを作ることができます．ここでは数個のデータの大きさを比較したいので，棒グラフを選んでみました．

画面のレイアウト

自分が手でグラフを書くときのことを思い出してください．まず，グラフ用紙に向かって，目盛を決めますね．コンピュータでグラフを描くときも，まず目盛を決めなくてはなりません．目盛を決めるときは，画面全体のレイアウトも同時に考えなくてはなりません．画面のレイアウトを考えて，目盛を決めてみましょう．

```
  0 1 2 ………                    28
0 <<グラフ>>
1 ─────────────────────────────
2  1 |◆……………………………………◆ 18
3  2 |◆…………………………………◆ 16
4  3 |◆……………◆     9
5  4 |◆………◆     6
6  5 |◆…◆     4
7 ─────────────────────────────
: ご゛うけい=  53
: へいきん=  10. 6
:
23
```

左の数字……………3ケタ
(数字1ケタ+符号1ケタ+区切り1ケタ)
区切り(縦線)………1ケタ
データ表示の数字…4ケタ
(数字2ケタ+符号1ケタ+区切り1ケタ)
グラフ………………20ケタ
残り…………………1ケタ
計　29ケタ

上の図のように，最初に〈グラフ〉とタイトルを書き，次の行に1本の線を引きます．グラフを描く部分は，一番左に入力した順序を表す番号を表示し，縦線を引いてから，特定のマークをデータの値に対応する数だけ並べて棒グラフとします．グラフの横には，データの値も表示しましょう．上の図のようなレイアウトだとグラフのために使えるケタ数は，

$$29-(3+1+4)=21$$

となりますが，少し余裕をみて20ケタ(20文字分)取っておくことにしましょう．この20ケタの中にすべての横棒を収めなくてはならないわけですが，そのためにはグラフで表す最大の値を求めて，その長さを20にしなければなりません．

最大値を求める

最大値を求める部分をプログラムにすると，次のリストのようになります．

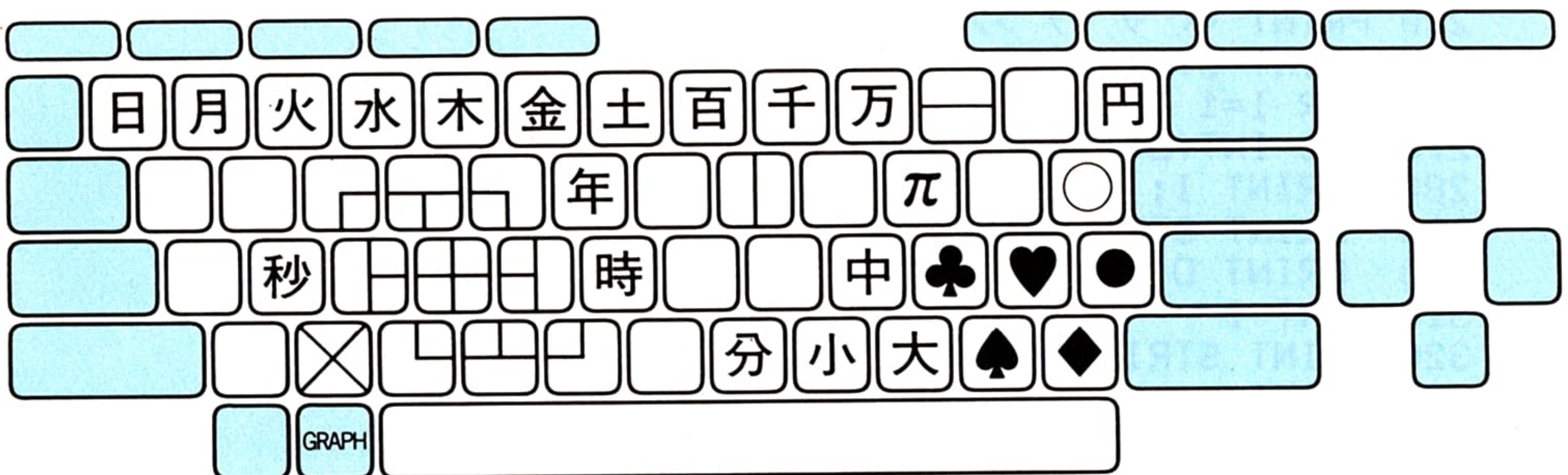
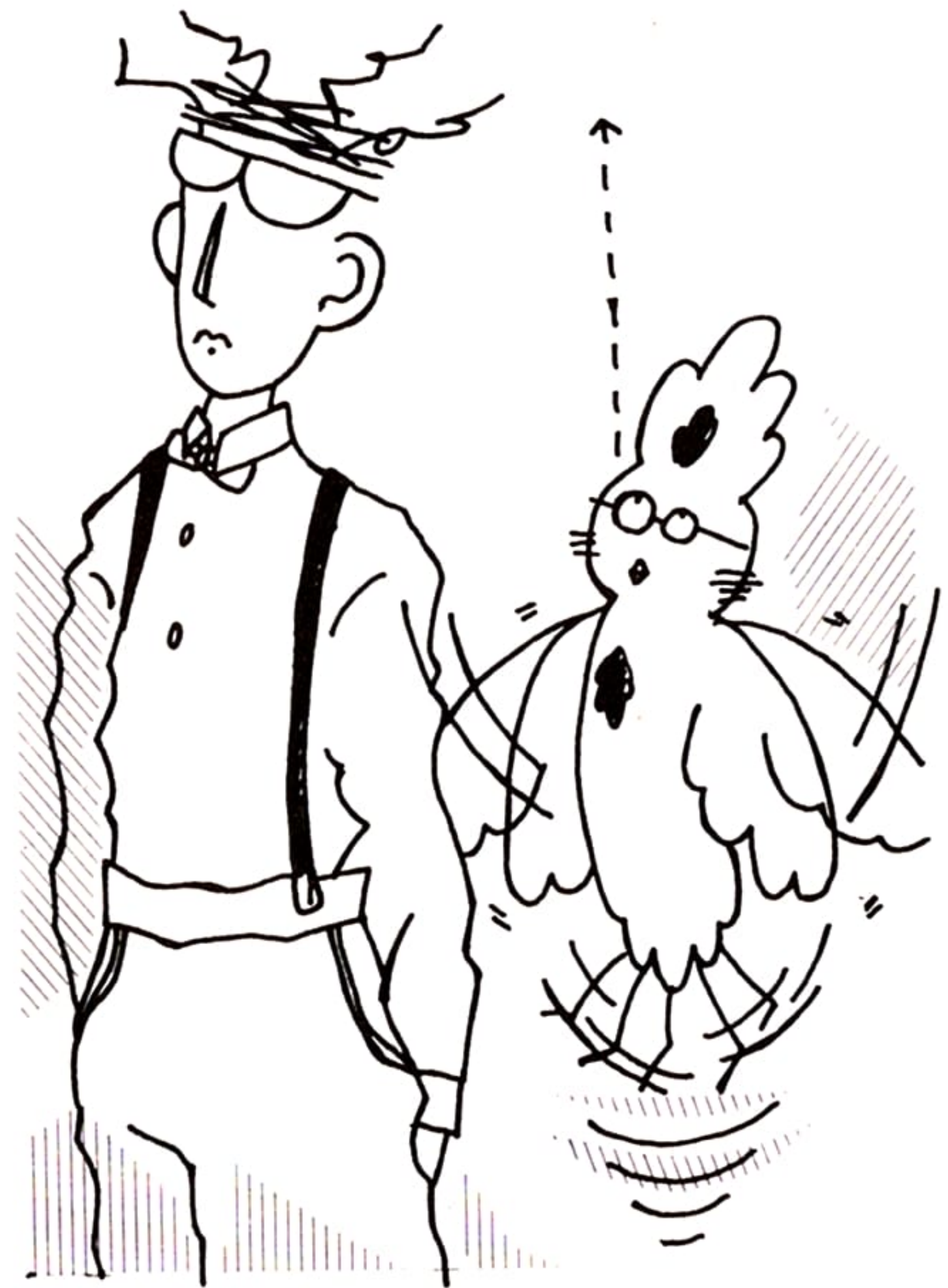
リスト

```
170
180 MX=D(1)
190 FOR I=2 TO N
200  IF MX<D(I) THEN MX=D(I)
210 NEXT I
```

最大値を表す変数としては，MXを用意しています．とりあえずD(1)の内容を最大値とみなしてMXの値とします．そしてこの最大値MXとD(I)を，Iが2からNまで比較してみます．つまりMXと，D(2)，D(3)……と，どちらが大きいかを比べるわけです．もしMXの方が小さいときは，MXより大きいD(I)の値を，新たなMXの内容とします．これをNまで繰り返すと，MXにはD(I)の中の最大値が入っていることになります．

最大値が求まれば，グラフのスケール(目盛)を決めることができます．最大のデータの長さを20にすればよいのですから，

$$\text{棒の長さ}=\frac{\text{データ}}{(\text{最大値}/20)}$$

で求めることができます．棒グラフを描くといっても，ここではそれらしいマークを上の式で求めた長さ分だけ並べて表示することにしています．MSXに用意された特殊な記号を打ち込むためには，GRAPHキーを押しながらそれぞれのキーを押します．ここでは◆マークを並べてグラフを描いてみることにしましょう．

GRAPH キーを押した状態のキーボードの配置図

グラフの完成

さきほど出てきた計算式の「棒の長さ」を表す変数をSCとすると，マークをSC個，横に並べて描けばよいことになります．このとき，

```
FOR J＝1  TO SC
  PRINT "◆";
NEXT J
```

としてもわるくありません．しかし，文字を横に並べて表示するための，専用の機能が用意されているので，これを使うことにしましょう．STRING$です．

```
STRING$ (□ , "□")
          ↓     ↓
       並べる数 並べる記号
```

この場合，◆マークをSC個並べて表示するためには，

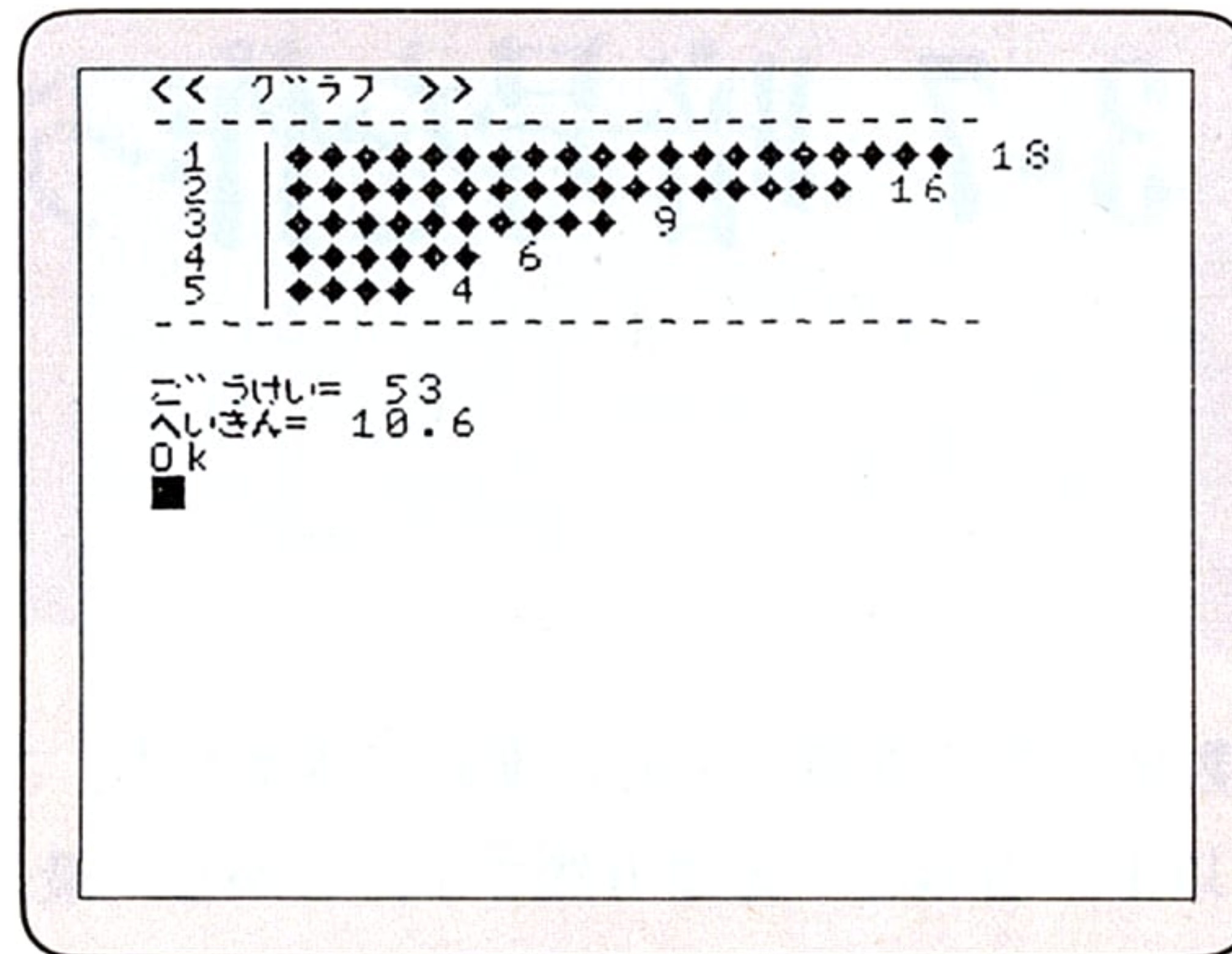
```
PRINT STRING$(SC , "◆")
```

とすればよいのです．グラフを描く部分のリストは，次のようになります．

リスト

```
220 '
230 CLS
240 PRINT"<< ｸﾞﾗﾌ >>"
250 PRINT STRING$(25,"-")
260 FOR I=1 TO N
270  SC=INT(D(I)/(MX/20))
280  PRINT I;"|";
290  PRINT STRING$(SC,"◆");
300  PRINT D(I)
310 NEXT I
320 PRINT STRING$(25,"-")
330 PRINT
340 PRINT"ごうけい=";S
350 PRINT"へいきん=";AV
360 END
```

実際に実行した画面を見てみましょう.

```
<< ｸﾞﾗﾌ >>
-------------------------
 1 |◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆ 18
 2 |◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆◆ 16
 3 |◆◆◆◆◆◆◆◆◆◆ 9
 4 |◆◆◆◆◆◆ 6
 5 |◆◆◆◆ 4
-------------------------

ごうけい= 53
へいきん= 10.6
Ok
■
```

意図したとおりのグラフが描けましたね．でも，このプログラムが万全であるわけではありません．個々のデータは 0 ～99の間でないと表示がくずれますし，データの個数にも制限があります．また，最大値MXとかけ離れた小さなデータがあると，◆マークが表示されないこともあります．

グラフに興味のある人は，この本を読み終わった頃に再度グラフに挑戦してください．きっと，もっとよいものができるでしょう．

最後に，全体のプログラムリストを載せておきますので，参考にしてください．

リスト

```
100 'ごうけい と へいきん
110 INPUT"にんずう=";N
115 DIM D(N)
120 FOR I=1 TO N
130  INPUT D(I)
140  S=S+D(I)
150 NEXT I
160 AV=S/N
170 '
180 MX=D(1)
190 FOR I=2 TO N
200  IF MX<D(I) THEN MX=D(I)
210 NEXT I
220 '
230 CLS
240 PRINT"<< ｸﾞﾗﾌ >>"
250 PRINT STRING$(25,"-")
260 FOR I=1 TO N
270  SC=INT(D(I)/(MX/20))
280  PRINT I;"|";
290  PRINT STRING$(SC,"◆");
300  PRINT D(I)
310 NEXT I
320 PRINT STRING$(25,"-")
330 PRINT
340 PRINT"ごうけい=";S
350 PRINT"へいきん=";AV
360 END
```

MSX BASIC

3-7 暗号を作ってみよう

ここまでのお話は，数値を扱う課題を中心に進めてきました．これは，MSXがコンピュータ(計算機)である以上，当然といえば当然です．しかし，最近のコンピュータの用途は多種多様で，コンピュータを〝電子計算機〟と呼ぶには，範囲が広がり過ぎています．コンピュータの重要な役割として計算があることは今でも変わりありませんが，最近では数値だけでなく，文字のデータを蓄えて，必要に応じてこれを加工するという機能も重視されています．

そこで，これからの3つのsectionでは暗号を作るという目標を掲げて，MSXの文字処理能力を探究してみようと思います．

ところで暗号とひとくちにいっても，いろいろな作り方があります．このsectionでは，もとの文の中に，その文に含まれている文字数と同じ数の文字を，適当に挟み込む方法で，暗号を作ってみます．挟み込む位置に規則性があるとすぐにバレてしまうので，挟み込む位置は乱数で決めることにします．挟み込む文字も，もちろん乱数で決めることにしましょう．

キョウ…………もとの文
　↓（適当な文字を3文字挟む）
キニョハカウ…暗号

文字を蓄える

まず，もとの文を蓄えなくてはなりません．これまで，文字を変数に蓄えることはしていませんでした．そこで，変数に文字が蓄えられるようにしなければなりません．文字を蓄えておく変数を文字変数と呼び，変数名に＄マークをつけることで普通の変数と区別します．

それではさっそくプログラムを作っていくことにしましょう．新しい機能を持った命令も出てきますが，文字処理だからといって特に難しいことはありません．

まず，もとの文を変数に蓄えておきましょう．

```
100  INPUT A$
```

もちろん，

```
100  INPUT "もとのぶん"; A$
```

のように，入力のときのガイドを入れることもできます．次に，挟み込む文字の数を決めるのですが，これはもとの文に含まれる文字の数と同じ，ということに決めてあります．もとの文が入っている文字変数A＄の中が何文字かを調べるためには，文字列の長さを調べる機能があるので，それを利用すればよいでしょう．LENは文字列の中に，濁点や空白なども含めて，何文字入っているかを調べてくれます．これは次のような型で使います．

〔1〕A＝LEN (A$)
〔2〕A＝LEN ("MSX")
} A$＝"MSX"とすると両方とも3という値が入る

これを使って，もとの文A＄の中に何文字入っているのかを調べる部分をプログラムにしてみると，

```
110  L＝LEN (A$)
```

となります．これで，LにA＄の中に含まれている文字数が入ることになります．

さて，いよいよ暗号化します．ここで例によって暗号化の手順を整理しましょう．

手順

① 文の長さを調べる
② 何文字目に文字を挟み込むかを決める
③ どの文字を挟み込むかを決める
④ ②で決めた位置を境に文を２つに分ける
⑤ ２つに分けた文の間に③で決めた文字を挟み込んで合成する
　①～⑤をL回(もとの文の文字数分)だけ繰り返す
⑥ 暗号文を表示する

これで暗号化の手順の整理はつきました．では，この手順をBASICで書き表していきましょう．

まず繰り返しの設定をします．繰り返しは，上で整理した手順①～⑤を，110行で調べたLの回数分行います．そうするとプログラムは，

```
120  FOR  I＝1  TO L
```

となります．次に，①～⑤をそれぞれBASICに書き直していきます．

文の長さを調べる

まず，文の長さを調べます．

```
130  LL＝LEN (A$)
```

A＄に文字を挟み込んでいくと，そのたびに文の長さが変わります．繰り返しの中で毎回文の長さを調べ，LLという変数に収めておきます．

挟み込む位置を決める

次に，文字を挟み込む位置を決めます．

```
140  N＝INT (RND (1)*LL＋1)
```

長さがLLの文字列A＄の中の，何文字目に文字を挟み込むかを決めて，Nにその数字を蓄えます．Nには１～LLの間のどれかの数字が入ります．

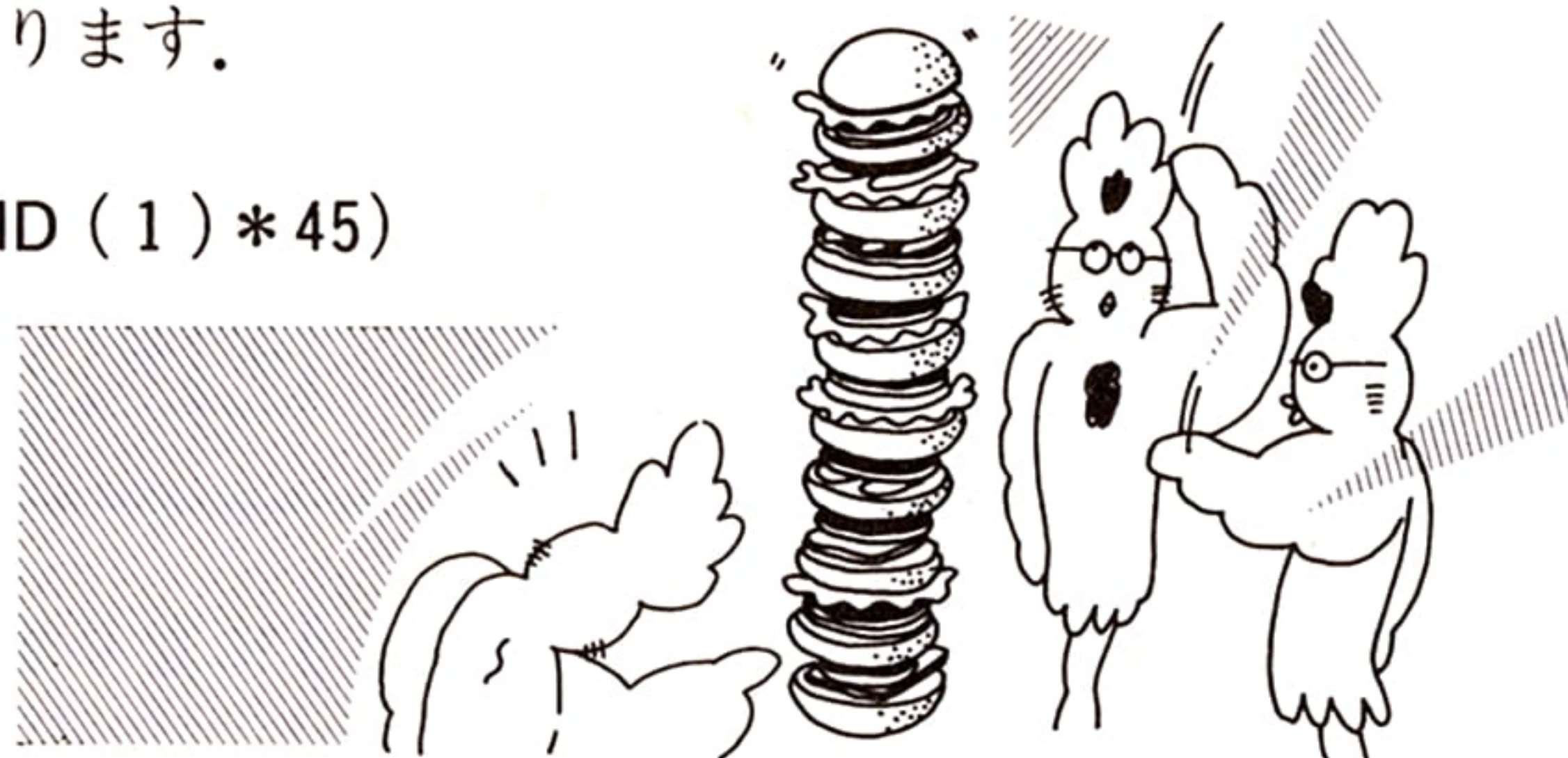
挟み込む文字を決める

①②はこれまでの知識だけでできましたが，③はそうはいきません．

実はBASICで扱うすべての文字には番号(コード)が付いているのです．たとえばａには97，アには177というようにです．このような文字を表す番号を，キャラクタコード(文字コード)と呼んでいます．どの文字を挟み込むかを決めるときには，乱数でこのキャラクタコードを指定してやればよいでしょう．キャラクタコードについては，巻末のAPPENDIXを参考にしてください．

ここで注意しなければならないことがあります．それは，文字を挟み込むときに，もとの文がカナ文字ばかりなのに挟み込む文字がアルファベットだったら，すぐに解読されてしまうことです．もとの文を入力するのはカタカナと決めて，挟み込む文字もカタカナにしておきましょう．

挟み込む文字のキャラクタコードをXとすると，Xにはカタカナ「ア」を示す177から「ン」の221の間のコードを代入します．そこで，適当なカタカナのキャラクタコードを求める部分は次のようになります．

```
150  X＝177＋INT (RND ( 1 )＊45)
```

150行を実行すると，Xには，挟み込む文字のキャラクタコードが入ります．次に，コードを実際の文字にしなければなりません．

キャラクタコードを文字にする命令は，CHR＄(　)です．これは，カッコの中のコードを表す文字を作るものです．たとえば，？CHR＄(177) RETURN とすれば，「ア」という文字が表示されます．

D＄という変数に，挟み込む文字を入れておきましょう．

```
160  D＄＝CHR＄ (X)
```

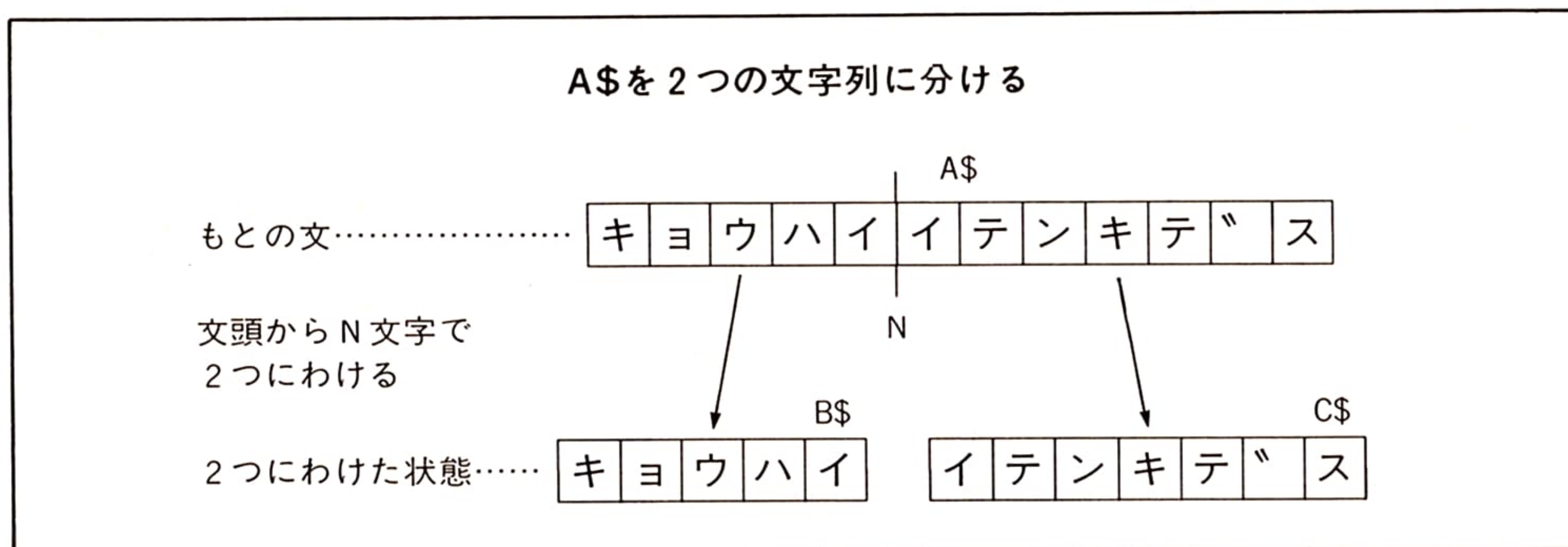
文を2つに分ける

文をどこから分けるかは②で決めてありますから，いよいよA＄を２つに分割してみましょう．文を２つに分けるためには，LEFT＄，RIGHT＄の２つを使います．LEFT＄は文字列の左から，RIGHT＄は文字列の右から，指定した数だけ文字を取り出します．

LEFT$(文字列，X)
　文字列の左からX字目までを取り出す
RIGHT$(文字列，X)
　文字列の右からX字目までを取り出す

この命令を使って，A＄を左からN文字目を境に，B＄，C＄の２つに分けます．図に示してみましょう．

A$を２つの文字列に分ける

A$
もとの文……………………　キ ョ ウ ハ イ | イ テ ン キ テ ゛ ス
N
文頭からＮ文字で
２つにわける
B$　C$
２つにわけた状態……　キ ョ ウ ハ イ　　イ テ ン キ テ ゛ ス

このような操作をプログラムにすると，次のようになります．

```
170　B＄＝LEFT＄(A＄，N)
180　C＄＝RIGHT＄(A＄，LL－N)
```

2つに分けた文の間に文字を挟む

文字列の結合には数字のたし算と同じように，＋の記号を使います．これを使えば，２つの文字列の間に文字を挟み込むのは簡単です．

```
190　A$＝B$＋D$＋C$
```

ここまでが，①～⑤で整理した一連の流れを，BASICで表現したものです．これをL回繰り返すので，繰り返しの範囲を示す，

```
200  NEXT  I
```

を付けておきましょう．

あとはできあがった暗号文を表示させるだけです．これは単純に，

```
210  PRINT  A$
```

とすればよいでしょう．

こうしてできたプログラムのリストをまとめてみると，次のようになります．新しいポイントがたくさん出てきた割には，短いリストになっています．

リスト

```
100 INPUT A$
110 L=LEN(A$)
120 FOR I=1 TO L
130  LL=LEN(A$)
140  N=INT(RND(1)*LL+1)
150  X=177+INT(RND(1)*45)
160  D$=CHR$(X)
170  B$=LEFT$(A$,N)
180  C$=RIGHT$(A$,LL-N)
190  A$=B$+D$+C$
200 NEXT I
210 PRINT A$
```

それでは，これで実際に暗号ができたかどうかを確かめてみましょう．

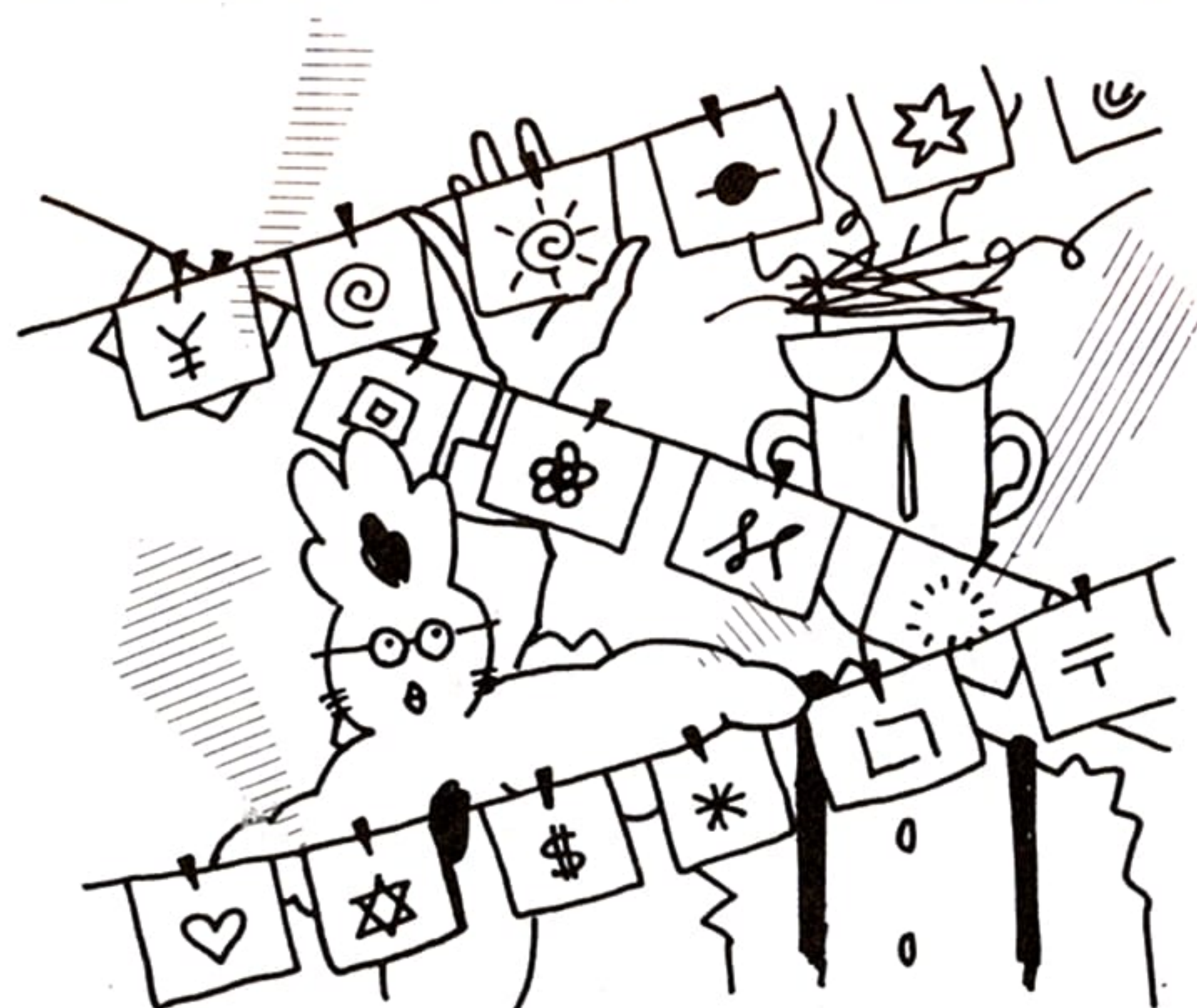
```
run
? ｷｮｳﾊｲｲﾃﾝｷﾃﾞｽ
ｷﾆｮｳﾊｶｲｲﾛﾃﾝﾓｵｷｲﾅﾆﾊｹﾃﾕﾇﾞｽ
Ok
■
```

MSX BASIC

3-8 暗号文の作成と解読

3－7では，比較的短くて，すっきりとした暗号作成のプログラムを作ることができました．この程度のものでも，暗号文だけパッと出されたら，そうそうすぐには解読できないでしょう．

しかし，このプログラムで満足しているわけにはいきません．さきほどのプログラムにも，まだまだ不満な点，改良すべき点があります．このような点のうち大きなものを挙げると，

① もとの文がカタカナに限られる
② 暗号を解読する部分がない

となります．MSXは，アルファベットの大文字，小文字，ひらがな，カタカナと多くの文字が使えるのに，カタカナしか使えないのはどうにもおもしろくありません．

また，暗号は解読できなければ単なるデタラメになってしまいます．解読するプログラムは，暗号化の逆をやればよいのですから，これを作れないことはありません．しかし，解読するためにはデータとして，

○ 暗号化された文
○ どこに文字を挟んだかというデータ

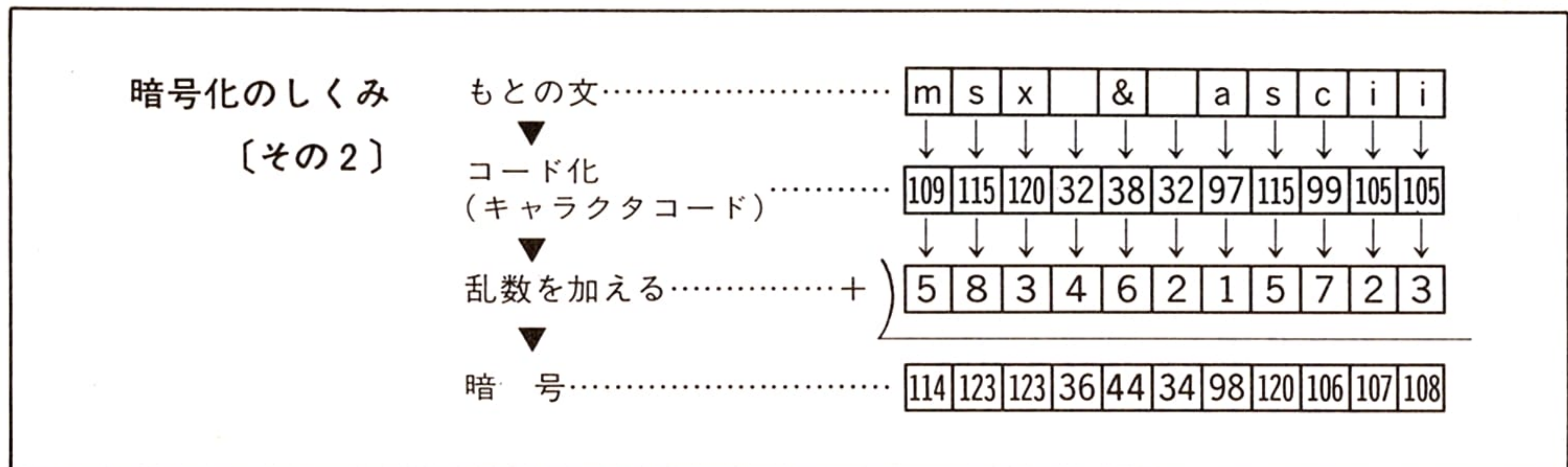
の2つが必要になります．暗号化された文そのものはいいとして，挟んだ文字の位置のデータが暗号を解くカギになっているのですが，このカギの量があまりにも多いのです．カギをどう保存するのか，どう相手に渡すのかを考えると，かなり難しそうです．

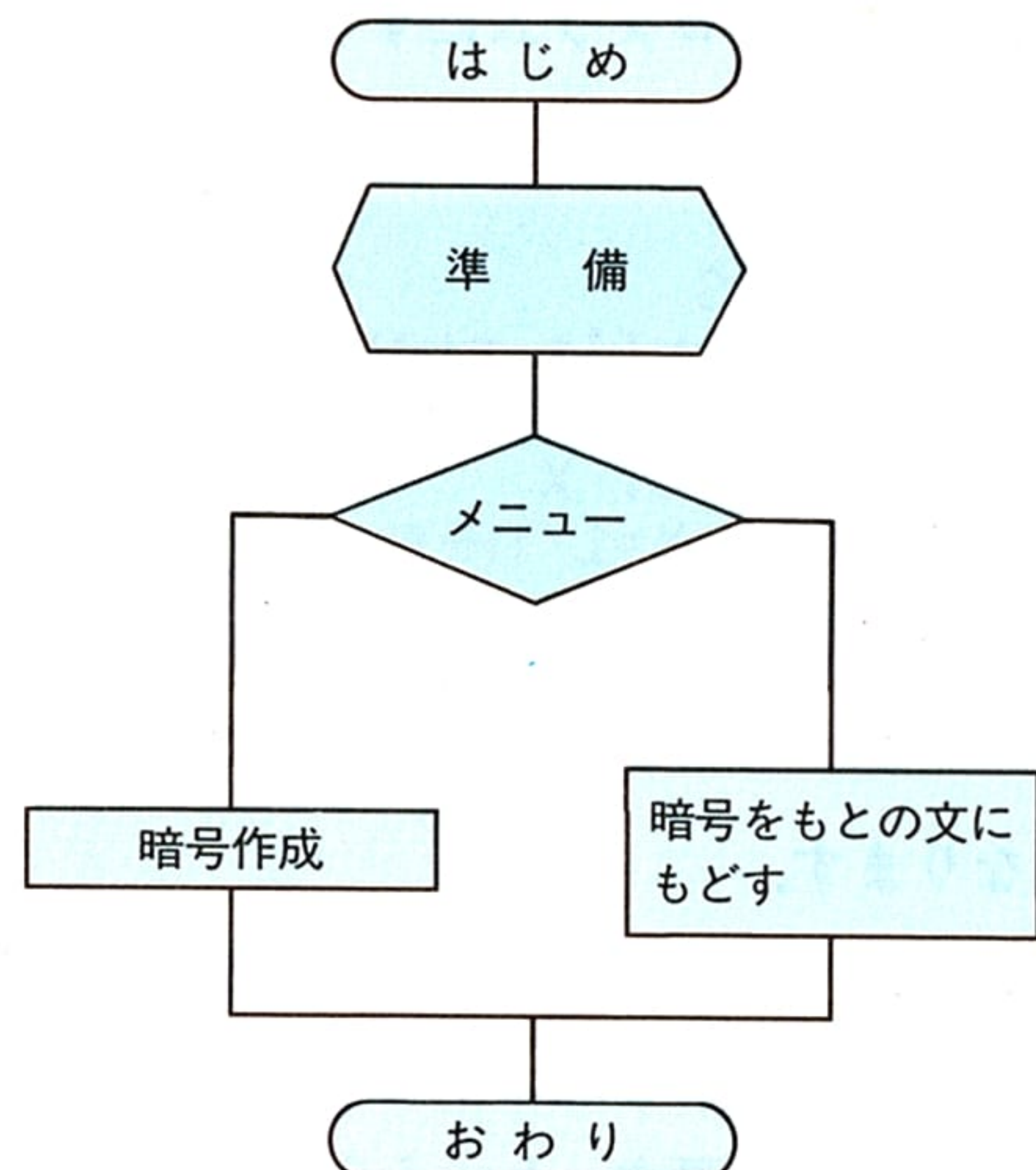
新しい暗号の規則

どうも行き詰まってしまったようです．思いきって暗号化の規則を変更しましょう．

新しい方法は，もとの文に文字を挟み込むものではなく，もとの文の各文字を，ある規則で直接変換するものです．文字を変換する規則のなかで，3-7に出てきたキャラクタコードをふたたび使います．まずもとの文の文字を，1文字ずつバラバラにして，バラバラにした文字をキャラクタコードに直します．キャラクタコードは数字ですから，＋－＊／などで，加工することができます．この演算に規則性があると，すぐに解かれてしまうので，ここでも乱数を利用して，0～9の範囲の適当な数を各文字のキャラクタコードに加えてやり，その結果を暗号としてみましょう．

暗号化のしくみ〔その2〕

もとの文	m	s	x		&		a	s	c	i	i
コード化（キャラクタコード）	109	115	120	32	38	32	97	115	99	105	105
乱数を加える ＋	5	8	3	4	6	2	1	5	7	2	3
暗号	114	123	123	36	44	34	98	120	106	107	108

この規則をまとめてみると，上の図のようになります．細かい点でまだ詰めなければならない部分はありますが，大筋はこれでよいでしょう．これならば，3-7で作った暗号のように，もとの文をカナ文字に限る必要はなく，原則としてどの文字でも暗号化することができます．

また，さきほどのプログラムには暗号を解読する部分がありませんでしたが，今度は最初からこれを組み込んでおくことにしましょう．このプログラム全体の構造は右の図のようになります．

ところでこれまでに作ったプログラムは，細かいレベルでいくつかの部分に分けられても，基本的には１つの仕事をする，１つのプログラムでした．しかし，ここで作るプログラムは大きく分けて，①普通の文を暗号にする，②暗号文をもとの文に戻す，という２つの仕事を１つのプログラムの中に含んでいます．そこで，どちらの仕事をさせるのかを選ぶ部分が必要になります．

これを一般にメニューと呼んでいます．メニューの機能はいくつかある仕事の中から，自分のさせたい仕事を選択することにあります．

それでは，実際に細かい点を整理しながらプログラムを作ってみましょう．

準備とメニュー

まずプログラムのタイトルを，プログラムの先頭に書いておきましょう．そして，暗号化した数字をしまっておく配列の宣言もします．この配列の名前は，Dとしておきましょう．

リスト

```
100 'あんごう
110 DIM D(255)
```

次にメニューの部分を作ります．ここで，暗号を作る仕事と，暗号を解読する仕事のうち，どちらを行うかを選ぶわけです．

メニューの働きを整理すると，（１）できる仕事の内容を表示する，（２）どの仕事を行うかをキーボードから入力する，（３）入力した指示に従って仕事を行う，となります．この部分をプログラムにすると，

リスト

```
120 CLS
130 PRINT"1:あんごう に する"
140 PRINT"2:かいどく"
150 INPUT X
160 IF X=1 THEN 180 ELSE 340
170 END
```

となります．

120行で画面を消したあと，130と140行で仕事の内容を表示します．150行でキーボードからどの仕事を選ぶかを番号で入力し，そこで１を選べば暗号作成，２を選べば暗号をもとの文に戻す，というようになるわけです．160行を見ると，今までにも何度か出て

きたIF～THEN～を使ってこの機能を実現しています．

ところで，そのあとにELSEという見慣れない文字が並んでいます．これは「さもなければ」という意味を表しています．これまでIF～THEN～を使うとき，条件が成立しているときは，THEN～を実行し，成立していないときは次の行を実行しました．ところがこのようにIF～THEN～ELSEの型で使用すると，条件が成立したときはTHEN～，しないときはELSE以下を実行するのです．

160行をELSEも含めて日本語で書くと，「もしXが1ならば(THEN)180行へ，さもなければ(ELSE)340行へ行け」ということになります．

130行，140行を見ると，一見2を選んだときだけ暗号解読の仕事をするように見えますが，実際には1以外が入力されれば，いつでも暗号解読を行うのがわかるでしょう．

このメニューのテクニックは，他のプログラムを作るときにも応用がきくので覚えておくとよいでしょう．

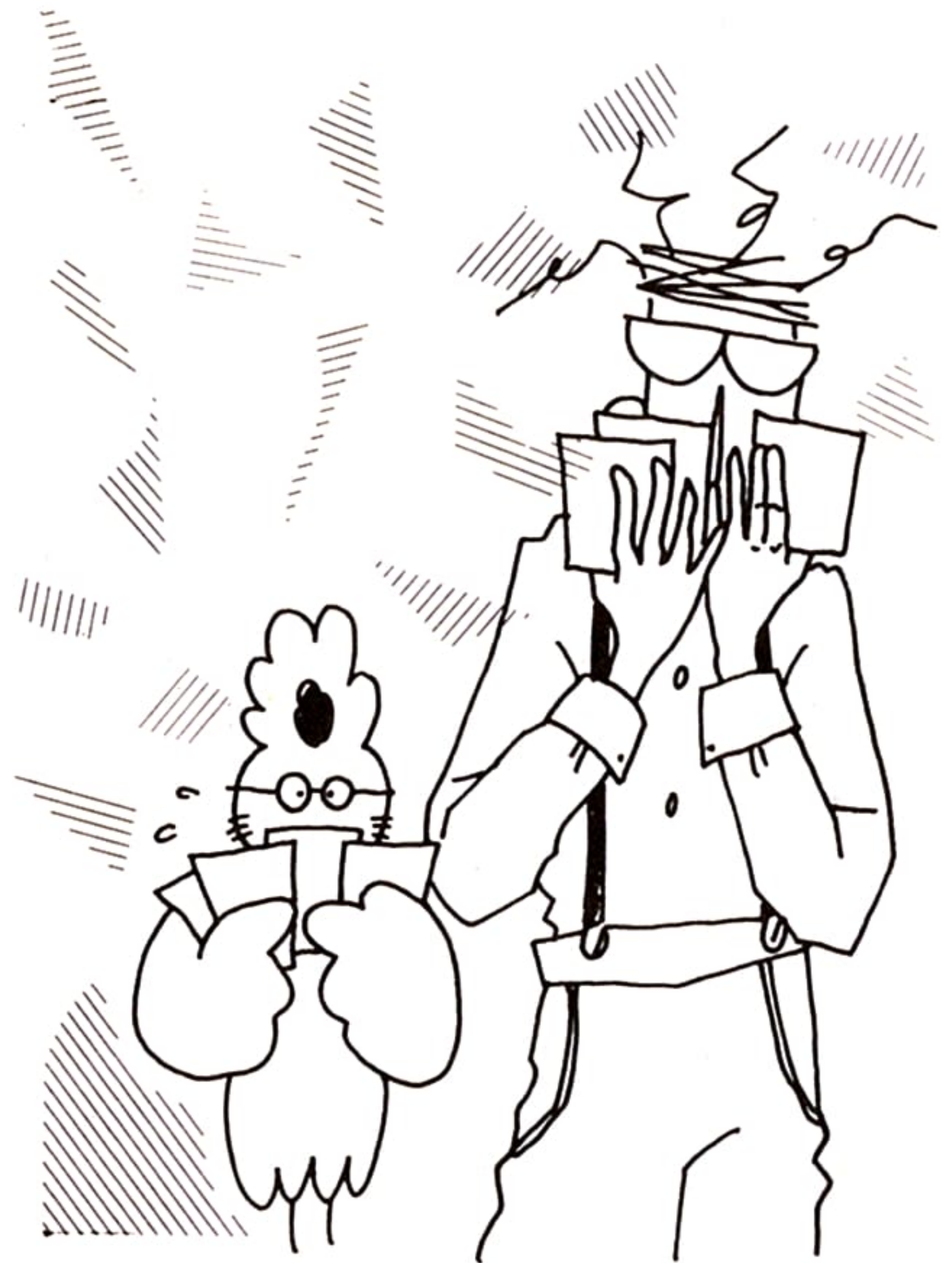
暗号を作る

次に，このプログラムの一番重要なところです．暗号化のだいたいの規則は，すでに紹介してありますが，もう一度細かい点まで確認してプログラムを作りましょう．

まず，もとの文を文字単位でバラバラにして文字コードにする部分はどうすればよいでしょうか．前の節で使ったLEFT＄，RIGHT＄を組み合わせて工夫すれば，なんとかなるかもしれません．

しかし，このようなときのために，MID＄という命令が用意されているので，これを使うことにしましょう．

MID$(文字列 , X , Y)

文字列の左からX番目の文字から，Y文字分の文字を取り出す

この命令を使うと，文字列の中の任意の１文字を取り出すことができます．左から１文字ずつMID＄を使って取り出せばよいでしょう．

MID＄を使ってもとの文を１文字１文字にバラバラにしたら，今度はそれをキャラクタコードに直さなければなりません．キャラクタコードを文字に直すのは，CHR＄でしたが，その逆もあります．それはASCです．

ASC(文字)

カッコの中の文字のキャラクタコードを求める

これで，もとの文を１文字ずつキャラクタコードに直すことができます．

次に乱数を加えるのですが，前にも触れたとおり，乱数は実行するたびに同じ順序(系列)で出ます．スロットマシンの例では，乱数の系列がデタラメの方がよかったので，X＝RND(－TIME)のようにして系列をランダムに設定しました．しかし，暗号の解読のことを考えると，全くデタラメでも困ります．銀行のキャッシュカードの暗証番号のように，ある数を知っていれば暗号が解けるようにしたいものです．

ところでRNDには，X＝RND(－X)の型で使うと，Xの値が同じなら，その後RND(１)で発生する乱数の系列を同じにする機能があります．ここではこの機能を利用して，Xを暗証番号にしましょう．

プログラムを見てください．

リスト

```
180 PRINT"もと の ぶ"ん: ";
190 LINE INPUT D$
200 PRINT
210 INPUT"あんしょう は"んこ"う";X
220 L=LEN(D$)
230 X=RND(-X)
240 FOR I=1 TO L
250  D(I)=ASC(MID$(D$,I,1))+INT(RND(1)*10)
260 NEXT I
270 PRINT"* もと の ぶ"ん * ";D$
280 PRINT:PRINT"* あんこ"う *"
290 FOR I=1 TO L
300  PRINT D(I);
310 NEXT I
320 END
```

190行ではもとの文を入力するのにINPUTではなく，LINE　INPUTを使っています．LINE　INPUTは，キーボードから文字を入れるための命令で，使い方はINPUTとあまり変わりませんが，(1)？が画面に表れない，(2) RETURN が押されるまで[,]などの記号が入っても入力される，(3)文字専用の入力である，といった特徴があります．入力する文の中に，[,]があると，INPUTでは入力の区切りとみなされて入力できなかったのですが，LINE　INPUT にはこのような制約がないので，こちらを使うことにしました．240行～260行で文をバラバラにして，キャラクタコードに直し，乱数を加えています．暗号化のエッセンスはこの3行だけです．

暗号化のエッセンス

```
250 D(I)=ASC(MID$(D$, I, 1))+INT(RND(1)*10)
```

- MID$(D$, I, 1) … 1文字取り出す
- ASC(…) → キャラクタコードにする
- +INT(RND(1)*10) → 0～9の乱数を加える

このように暗号化したあとは，もとの文と暗号にした数字を表示してこの部分は終わりです．

暗号の解読

解読の部分は，暗号を作る部分のまったく逆をやればよいのです．この部分のリストを載せておきましょう．

350行～380行で暗号文の文字数と暗号文を，390行で暗証番号を読み込んで，400～450行でもとの文に戻しています．まずD＄をカラにして，そのあと1文字ずつ解読した部分を足して，もとの文に戻していきます．460行～480行でもとに戻した文を表示してプログラムは終わりになります．

さあ，これでうまくいくでしょうか．次のsectionで実際に実行し，必要があればさらに改良してみましょう．

リスト

```
330 '
340 PRINT"* あんこ"うぶ"ん *"
350 INPUT"もじ"すう";L
360 FOR I=1 TO L
370  INPUT D(I)
380 NEXT I
390 INPUT"あんしょう ば"んこ"う";X
400 D$=""
410 X=RND(-X)
420 FOR I=1 TO L
430  Y=D(I)-INT(RND(1)*10)
440  D$=D$+CHR$(Y)
450 NEXT I
460 PRINT
470 PRINT"* もと の ぶ"ん * "
480 PRINT:PRINT D$
490 END
```

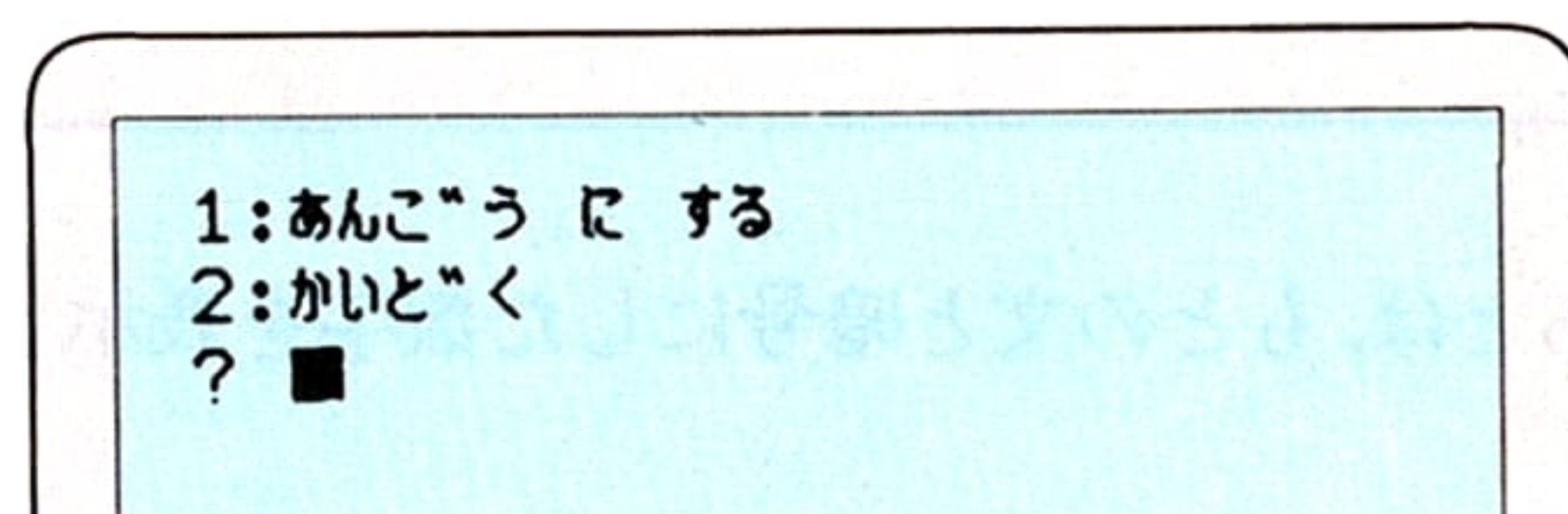
MSX BASIC

3-9 暗号テープを作って送ろう

3－8で作った暗号プログラムをさっそく試してみることにしましょう．

暗号を作成する

F・5 でプログラムを実行すると，

```
1:あんこ゛う に する
2:かいと゛く
? ■
```

のような画面となります．これがメニューです．まず普通の文を暗号にしてみましょう．1 RETURN として暗号にする部分を選びます．すると画面には，

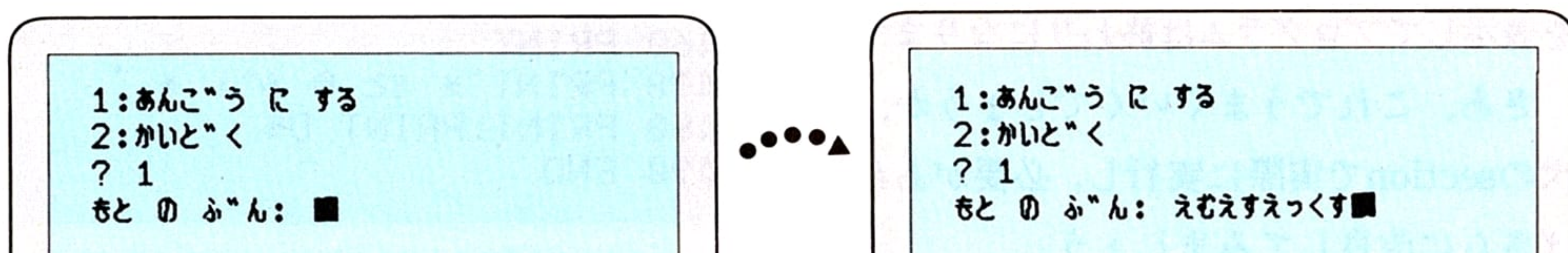
もと　の　ぶん：

と表示されるので，暗号にする前のもとの文をキーボードから打ち込みます．このとき使える文字は，アルファベットの大文字，小文字，ひらがな，カタカナ，そして数字と一部の記号です．グラフィック文字や漢字は使えません（巻末のキャラクタコード表(2)の解説を参照してください）．

ここでは，次のような文を入れてみましょう．

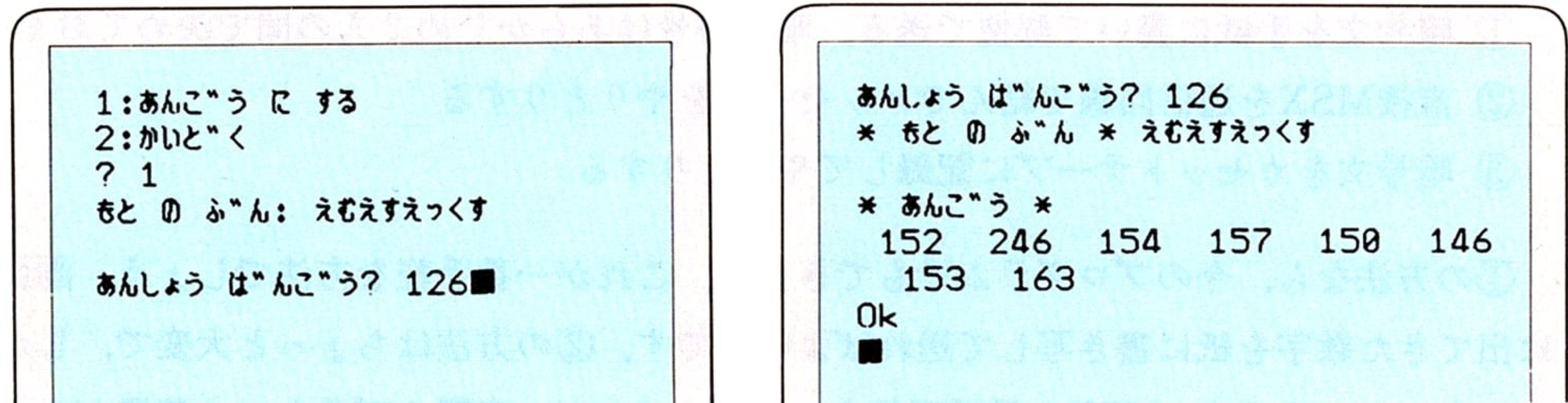
```
1:あんこ゛う に する
2:かいと゛く
? 1
もと の ふ゛ん: ■
```

……▲

```
1:あんこ゛う に する
2:かいと゛く
? 1
もと の ふ゛ん: えむえすえっくす■
```

続いて暗証番号を聞いてきます．1以上の数を入れます．とりあえず126としておきます．この番号を忘れると，二度ともとの文には戻らないのでよく覚えておいてください．

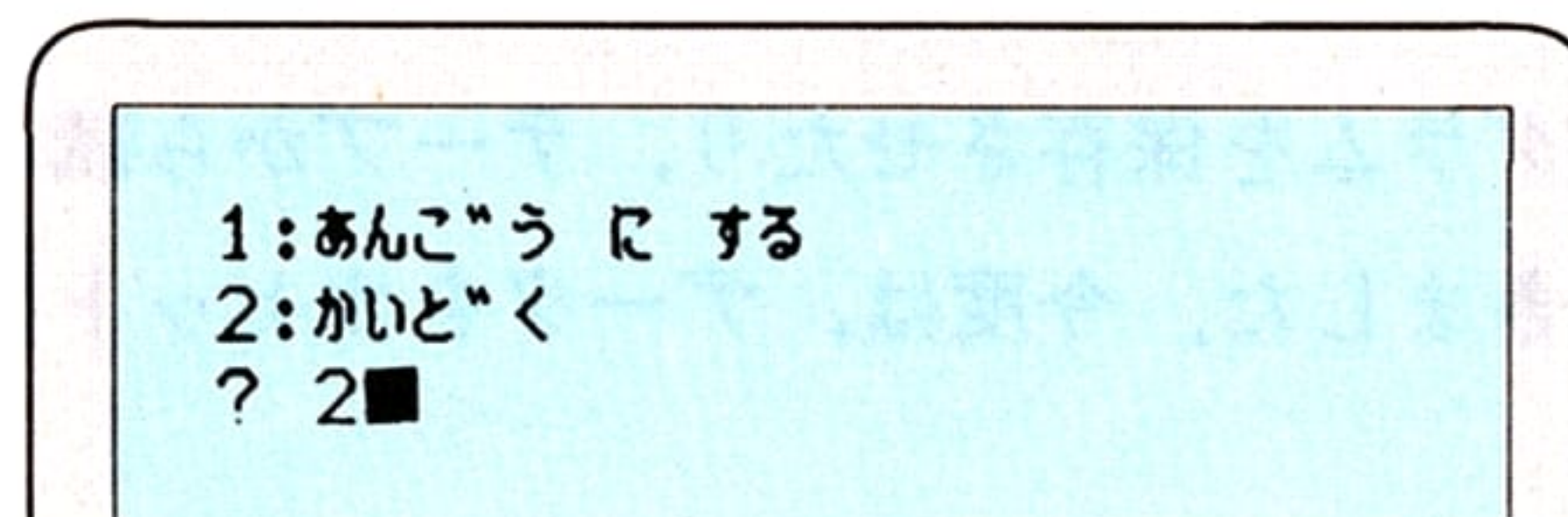
すると，次のようにもとの文と，暗号となった数字が表示されます．暗号の数字は紙にメモしておきましょう．

```
1:あんこ゛う に する
2:かいと゛く
? 1
もと の ふ゛ん: えむえすえっくす

あんしょう ば゛んこ゛う? 126■
```

```
あんしょう ば゛んこ゛う? 126
* もと の ふ゛ん * えむえすえっくす

* あんこ゛う *
 152  246  154  157  150  146
  153  163
Ok
■
```

暗号を解読する

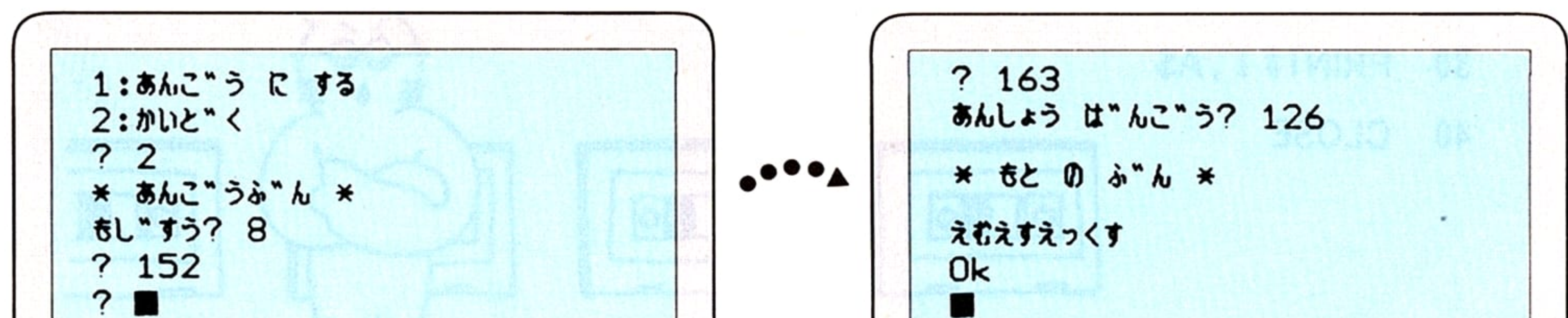
このようにして作った暗号を，今度は解読してみましょう．

F・5 でプログラムを実行すると，さきほどと同じようにメニューが表れてきます．今度は2を選びます．

```
1:あんこ゛う に する
2:かいと゛く
? 2■
```

するともとの文の文字数と，暗号文（数字）を入力するように求めてきます．さきほどの「えむえすえっくす」は8文字ですから，8 RETURN，各暗号数字は，152，246……ですから，152 RETURN，246 RETURN，……と打ち込んでいきます．暗号をすべて入れ終わったら暗証番号を入れます．この番号は暗号を作った時と同じものにします．暗証番号が合っていれば，無事にもとの文に戻ります．

```
1:あんこ゛う に する
2:かいと゛く
? 2
* あんこ゛うふ゛ん *
もじ゛すう? 8
? 152
? ■
```

●●●●▲

```
? 163
あんしょう ば゛んこ゛う? 126

* もと の ふ゛ん *

えむえすえっくす
Ok
■
```

暗号のやり取り

自分ひとりで暗号を作ったり，それをもとに戻したりしていてもおもしろくありません．誰かとやりとりする方法はないでしょうか．考えられる方法を挙げてみましょう．

① 暗号文を手紙に書いて郵便で送る．暗証番号はあらかじめ2人の間で決めておく
② 直接MSXを通信回線で結んでメッセージをやりとりする
③ 暗号文をカセットテープに記録してやりとりする

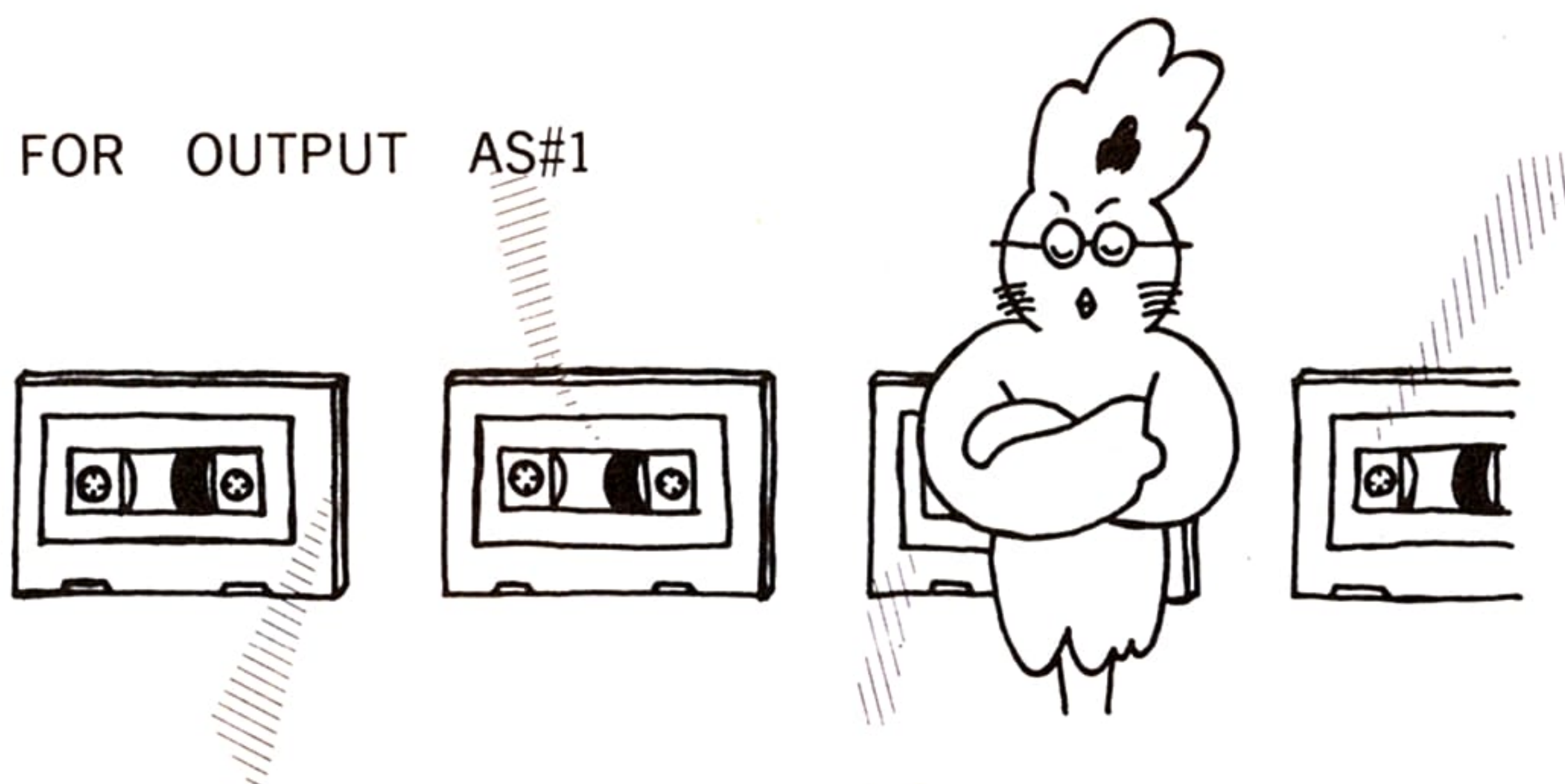
①の方法なら，今のプログラムでもできます．これが一番手軽な方法でしょう．画面に出てきた数字を紙に書き写して送ればよいのです．②の方法はちょっと大変で，しかもお金がかかります．MSXと電話回線をつなぐためには，音響カプラという装置が必要です．この音響カプラは値段も高く，まだ一般に普及していないので，今回はパスしましょう．

③の方法なら，実はこのプログラムをちょっと手直しすることで実現します．コンピュータのデータが記録されたテープでメッセージをやりとりするなんて，なかなかイマッぽくて楽しいでしょう．

カセットテープにプログラムを保存させたり，テープから読み込んだり，ということはもうすでに何回もしてきました．今度は，データをカセットテープに記録したり読み込んだりするわけです．

カセットテープにデータを記録

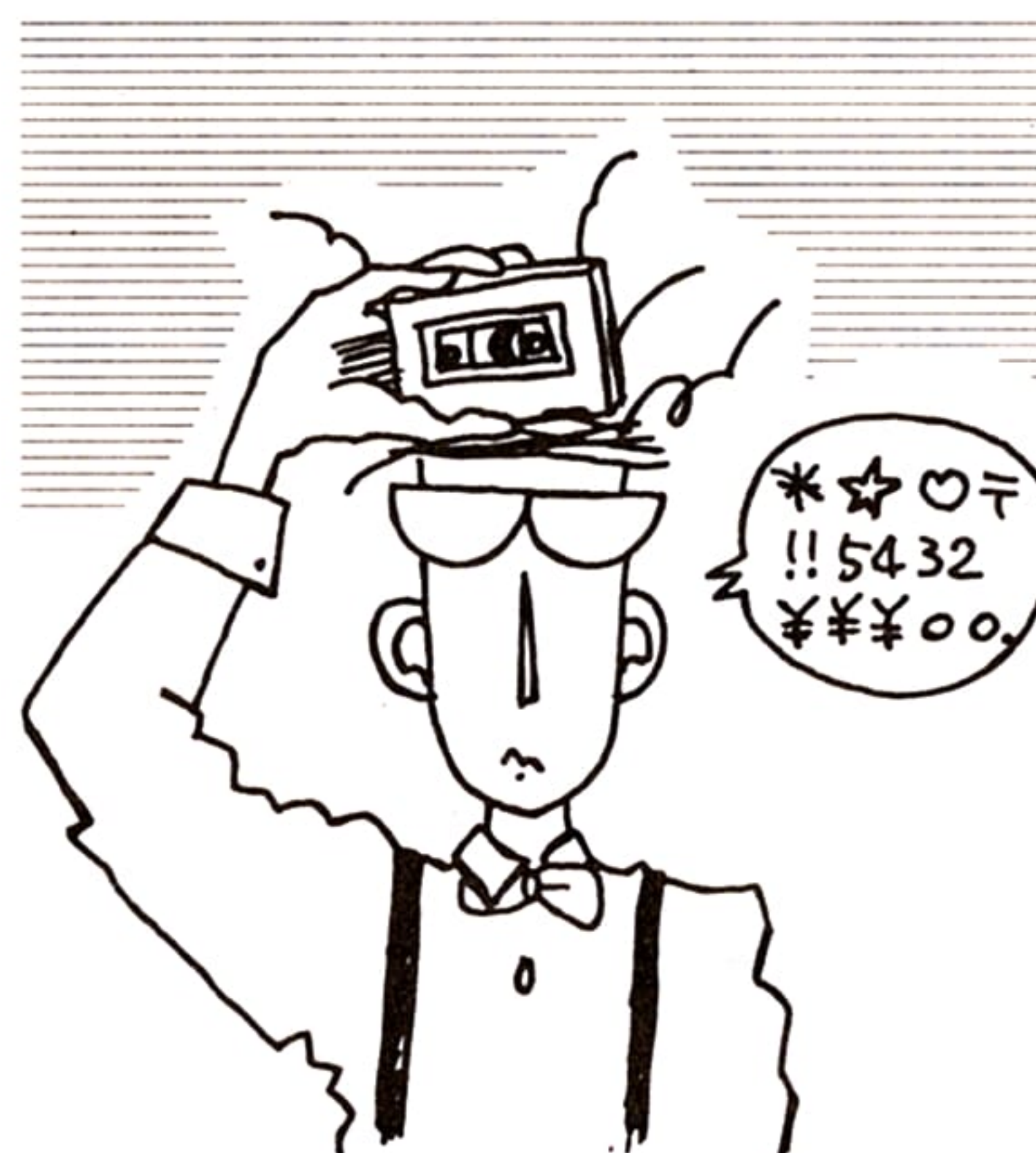
カセットへ記録するのも書くことには変わりないので，PRINTを使います．が，違うところもあるので，下に文字列をカセットに記録するプログラムの例を載せておきます．

```
10  INPUT  A$
20  OPEN "CAS:" FOR  OUTPUT  AS#1
30  PRINT#1,A$
40  CLOSE
```

OPEN "CAS：" FOR　OUTPUT　AS#1 というのが，「これからカセットテープに書き込みますよ」という一種の決まり文句です．30行でPRINT#1，という命令がありますが，これがカセットにデータを書き込む命令です．使い方は普通のPRINTとほとんど同じです．最後にCLOSEとあるのは，「これでカセットの使用を終わります」という決まり文句です．このように，最初にOPEN，終わりにCLOSEを付け，その間でPRINT#1を使ってカセットテープにデータを書くのです．

カセットテープからデータを読む

カセットにデータを書き込むときと同じように，OPENとCLOSEで挟まれた間で，INPUTの変形であるINPUT#1，を使って読み込みます．

```
10  OPEN "CAS:" FOR  INPUT  AS#1
20  INPUT#1,A$
30  CLOSE
40  PRINT  A$
```

10行のOPENの形が，書き込み用のものと少し違っていて，FOR　INPUTになっています．

プログラムの変更

カセットテープにデータを読み書きする基本はたったこれだけです．少し乱暴かもしれませんが，さっそくテープを使えるようにプログラムを手直ししてみましょう．まず暗号文を画面に表示させるときに，同時にカセットテープにも暗号を書き込むようにしています．そのために，285行にカセットを使う前の決まり文句を加えます．

```
285  OPEN "cas:" FOR  OUTPUT  AS#1
```

そして，あとで解読をするときに便利なように，もとの文の文字数をテープの先頭に書き込んでおきましょう．

```
287  PRINT#1 , L
```

暗号文を，実際にテープに書いていくためには，

```
305  PRINT#1 , D(I)
```

を付け加えます．300行では画面に，305行ではテープに書いていくのです．テープへの書き込みが終わったら，

```
315  CLOSE
```

を付けておきます．

次に読み込みの部分です．キーボードから手で打ち込むのは大変なので，入力の部分を，完全にカセットからの入力に置き換えてしまいましょう．この部分を手直しすると，次のようになります．

```
345  OPEN "cas:" FOR  INPUT  AS#1
350  INPUT#1 , L
360  FOR  I=1  TO  L
370   INPUT#1 , D(I)
380  NEXT  I
385  CLOSE
```

OPENとCLOSEが加わっただけで，基本的にはほとんど変わっていませんね．プログラムの変更はたったこれだけです．全体のリストを載せておきます．

リスト

```
100 'あんこ"う                          ①準備の部分
110 DIM D(255)
120 CLS                                ②メニューの部分
130 PRINT"1:あんこ"う に する"
140 PRINT"2:かいと"く"
150 INPUT X
160 IF X=1 THEN 180 ELSE 340
170 END
180 PRINT"もと の ぶ"ん: ";
190 LINE INPUT D$
```

```
200 PRINT
210 INPUT"あんしょう は"んこ"う";X
220 L=LEN(D$)
230 X=RND(-X)
240 FOR I=1 TO L
250  D(I)=ASC(MID$(D$,I,1))+INT(RND(1)*10)
260 NEXT I
270 PRINT"* もと の ふ"ん * ";D$
280 PRINT:PRINT"* あんこ"う *"
285 OPEN"cas:" FOR OUTPUT AS#1
287 PRINT#1,L
290 FOR I=1 TO L
300  PRINT D(I);
305  PRINT#1,D(I)
310 NEXT I
315 CLOSE
320 END
330 '
340 PRINT"* あんこ"うふ"ん *"
345 OPEN"cas:" FOR INPUT AS#1
350 INPUT#1,L
360 FOR I=1 TO L
370  INPUT#1,D(I)
380 NEXT I
385 CLOSE
390 INPUT"あんしょう は"んこ"う";X
400 D$=""
410 X=RND(-X)
420 FOR I=1 TO L
430  Y=D(I)-INT(RND(1)*10)
440  D$=D$+CHR$(Y)
450 NEXT I
460 PRINT
470 PRINT"* もと の ふ"ん *"
480 PRINT:PRINT D$
490 END
```

③暗号化の部分

④解読の部分

プログラムの使い方は特に説明するまでもないと思います．MSXとカセットテープレコーダとをケーブルでつないで，暗号にするときは録音の状態に，暗号をもとの文に戻すときは再生の状態にしておけばよいのです．

これでプログラムは完成です．このプログラムで暗号をテープに書き込めば，MSXを持っている友達と暗号テープのやり取りをして遊べます．渡す方法は，手渡しするのもよいでしょう．マイクロカセットを使えば，封筒に入れて送ることもできそうですね．

COLUMN

プリンタを使いたい人のために

ＭＳＸで，データや計算結果を表やグラフに表したいとき困るのが画面の一行に表示できる文字数が少ないということです．また，プログラムが長くなると，画面のうえだけで修正作業を行うのはずいぶんと骨が折れます．

このような場合，プリンタがあると便利です．プリンタは計算した結果などを画面にではなく，紙の上に書くものです．横80文字くらい書くことができますし，縦に関しては，紙がなくなるまで書き続けられます．

プリンタとＭＳＸは，ほとんどの機種でそのままつなぐことができますが，プリンタによっては，ひらがなの出ないものもありますので，「ユーザーズマニュアル」で確認してみてください．

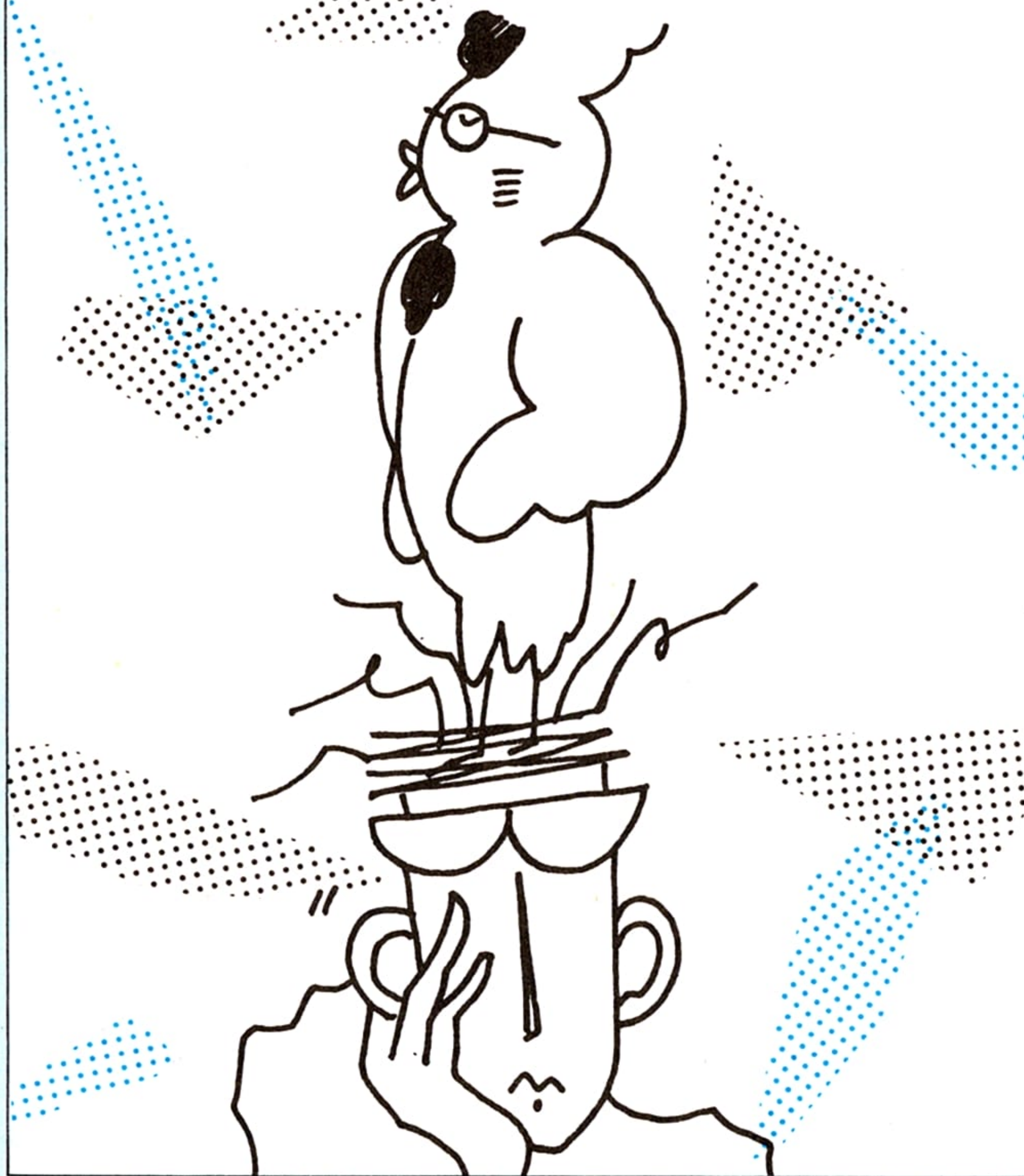
プリンタに何かを出力するための命令はいたって簡単です．これまで画面に表示する命令としてＰＲＩＮＴを使ってきましたが，プリンタに出力するためにはＬＰＲＩＮＴを使えばよいのです．Ｌが頭に付いたこと以外は，使い方，書式などはＰＲＩＮＴと変りません．

例）print “MSX”
　　lprint “MSX”

また，プログラムのチェック修正にプリンタを使いたいときは，ＬＩＳＴの頭にＬを付けて，ＬＬＩＳＴという命令を使います．使い方はＬＩＳＴとまったく同じです．

CHAPTER 4

グラフィック&サウンド

―― ゲーム・プログラミングへの招待 ――

4-1 MSXグラフィック入門

MSX BASIC

これまで，ひととおりBASICの基本的な機能を見てきました．ほんのわずかな間にプログラムが長くなり，少し戸惑っている人もいるかもしれません．命令の使い方などで少しでも不安があったら，こまめに「MSX BASICリファレンスマニュアル」などを引いてみることです．

さて，これからしばらくはこれまでと趣を変えて，MSXの特徴を活かしたグラフィック機能や，音楽の機能を中心に，楽しめるプログラムを作っていくことにしましょう．

2つの画面モード

これまで，MSXの画面についてきちんとした説明はしてきませんでした．ここで画面のモードということについて少し説明しておきましょう．

MSXの画面には，絵や図を描くグラフィックモードと，文字や数字を表示するテキストモードの2つがあります．グラフィックモードは，これまでの説明では「絵を描く画面」と言ってきたもので，細かい点を組み合わせて絵や図を描くことができますが，文字や数字の表示には工夫が必要です．また，テキストモードでは，リストを表示したり，計算の結果などを画面に表示したりすることはできますが，絵や図を描くことはできません．テキストモード，グラフィックモードは，表示文字数や解像度の違いによりさらに2つに分かれます．これを整理したものが次のページの表です．

4つある画面のうち，どれを使うかを指定するのがSCREEN命令です．CHAPTER 2では「SCREEN　2は絵を描くおまじない」というように説明していましたが，これはグラフィックの高解像度の画面を使う，という意味の命令だったのです．電源を入れた時点でMSXは，自動的にSCREEN 1にセットされているのですが，テキスト画面には絵を描くことができないので，SCREEN 2と指定して円を描いたのです．

モード		解像度	表示文字数	スプライト	特　　徴	指　定
テキスト	I	―――	最大40×24 (39×23)*	不　可	一画面に多くの文字が表示できる	SCREEN 0
	II	―――	最大32×24 (29×24)*	可	電源をいれたときの状態．SCREENで特に指定しないときはこの状態になっている	SCREEN 1
グラフィック	高解像度	256 × 192	―――	可	細かい点を組み合せて絵やデザインがかける	SCREEN 2
	マルチカラー	64 × 48	―――	可	点はあらいが一つひとつの点に色を付けられる	SCREEN 3

＊初期設定状態の表示文字数

MSXで表示できる画面の種類

また，プログラムの中でSCREEN 2と設定しても，プログラムの中でINPUT命令が出てきたり，プログラムが終わると，SCREEN 1に戻ります．通常はテキストモードの状態でプログラムを打ち込んだりリストを表示したりしますから，プログラムが終了すると同時に，SCREEN 1の状態に戻るのです．

SCREEN 0の状態は，一画面に多くの文字が表示できますが，あとで説明するスプライトが使えないなど，デメリットもあります．

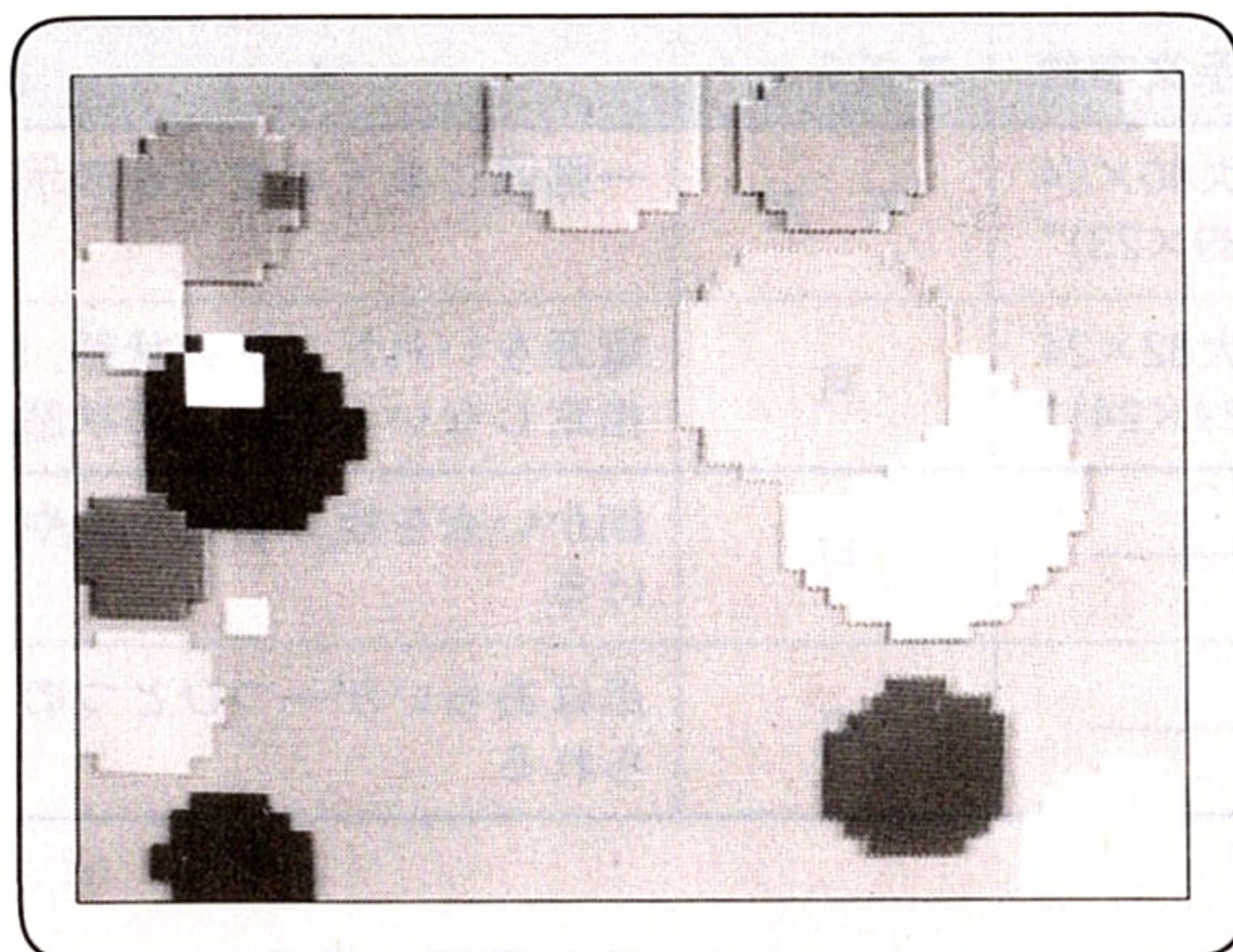
マルチカラーモード

2-5の最後に出てきたプログラムを，もう一度見てください．

リスト

```
10 X=RND(-TIME)
20 SCREEN 2
30 X=INT(RND(1)*256)
40 Y=INT(RND(1)*192)
50 R=INT(RND(1)*30+1)
60 C=INT(RND(1)*15+1)
70 CIRCLE(X,Y),R,C
80 PAINT(X,Y),C
90 GOTO 30
```

このプログラムは，20行のSCREEN命令により高解像度のグラフィックモードを指定しています．もう一度このプログラムを動かしてみてください．よくよく見ると，色がにじんで，円がちょっとおかしくなっているのがわかります．高解像度のグラフィックでは，一つひとつの点を単位として色を付けることができないので，どうしてもこうなってしまうのです．

一つひとつの点ごとに色を付けたいときは，SCREEN 3のマルチカラーモードを使います．20行をSCREEN 3に変えて実行してみてください．

確かに色はちゃんと付いていますが，点の大きさが大き過ぎるのが難点です．普段はSCREEN 1のテキストモードと，SCREEN 2のグラフィックモードを使うことの方が多いでしょう．

グラフィックモードとテキストモードの違いは，だいたいわかったと思います．最後にテキストモードでしか使えない命令，グラフィックモードでしか使えない命令がありますので，それをまとめておきます．

モード	そのモードしか使えない命令		
テキスト	INPUT	PRINT	LOCATE
グラフィック	CIRCLE	LINE	DRAW
	PSET	PRESET	PAINT

モードと命令の関係

まだ説明していないグラフィックの命令についても，順に解説します．

MSXグラフィックスの特徴

グラフィックの例は，すでにCHAPTER 2でも見てきましたが，機能の紹介を兼ねて，MSXグラフィックスの特徴をまとめておきます．

〔1〕16色のカラーが使える

これまで使ってきてわかるとおり，最高16の色を画面に表示することができます．

〔2〕スプライトが使える

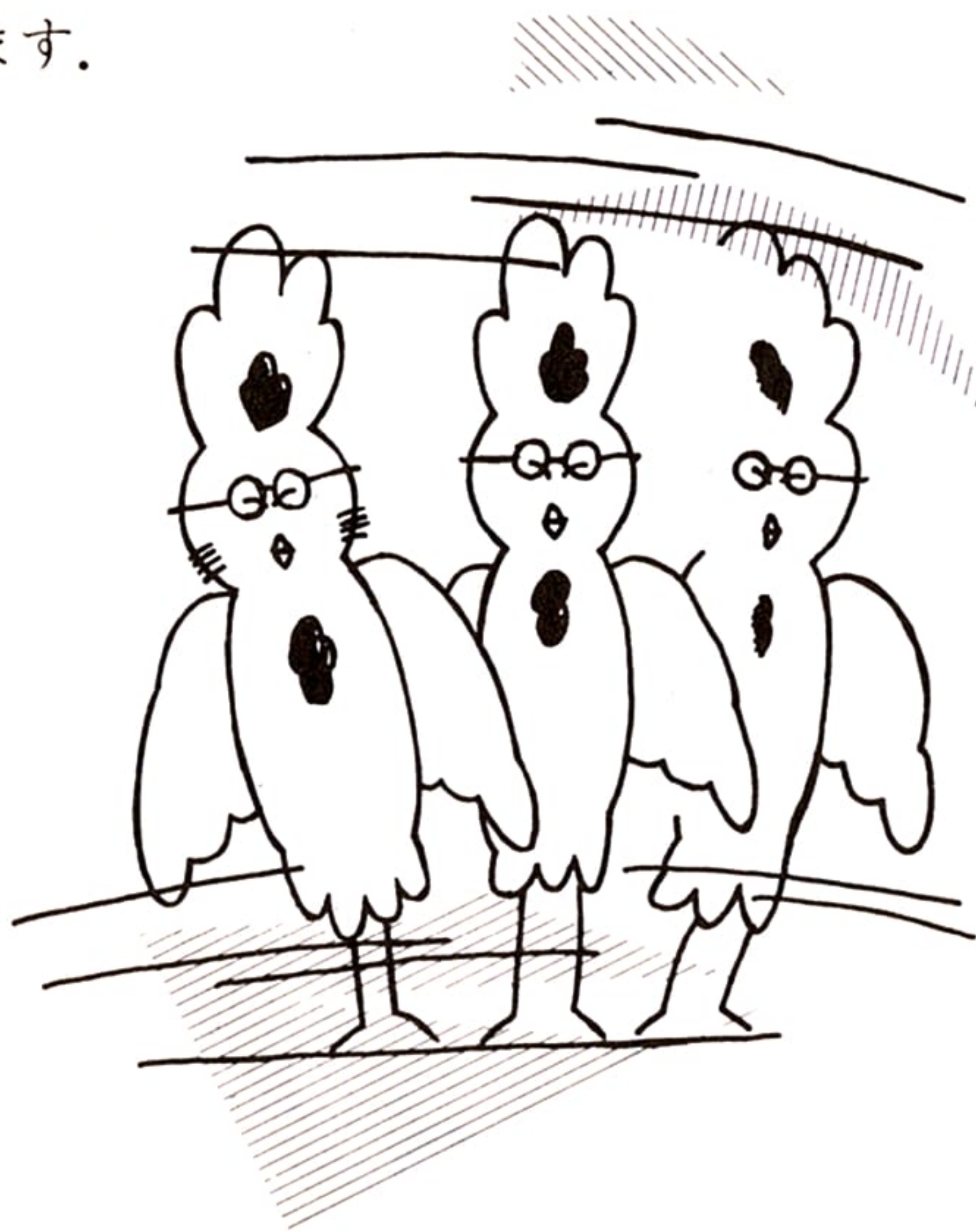
コマーシャルなどでコンピュータを使ったアニメーションを見る機会も増えてきました．自分でそういったアニメーションが作れれば楽しいですね．またゲームを作るときにパターンが簡単に動かせれば便利です．

このような夢を簡単にかなえてくれるのが，MSXのスプライト機能です．これは自分で絵を自由にデザインして，それを表示するだけでなく，画面の上を自由自在に動かすことができる機能です．スプライトについては，4－2から4－4で解説します．

〔3〕豊富なグラフィック命令

スプライト機能は，動く絵を描くのに便利なものですが，特に動きを必要としない絵を描くためにも，命令が豊富に用意されています．

○CIRCLE ………………円を描く
○LINE …………………線を引く
○PSET …………………点を打つ
○PRESET ………………点を消す
○PAINT ………………色を塗る

上に挙げたのは，その中の主なものです．CIRCLEやPAINTはすでに見てきましたが，それ以外の命令についても段々に解説していきます．

〔4〕DRAW命令

円を描くとき説明したように，グラフィック画面を使うときには，絵を描く位置を横，縦の座標で表します．しかし，MSXには座標などを使わなくとも，あたかもペンを動かすような感覚で絵が描けるDRAWという命令が用意されています．このDRAW命令は4－5で使ってみましょう．

これらの特徴は自分で実際にプログラムを打ち込み，結果を画面で確認してみた方が，ただ本を読むだけよりもはるかによくわかると思います．これからしばらく，これらの機能を具体的に説明していきます．

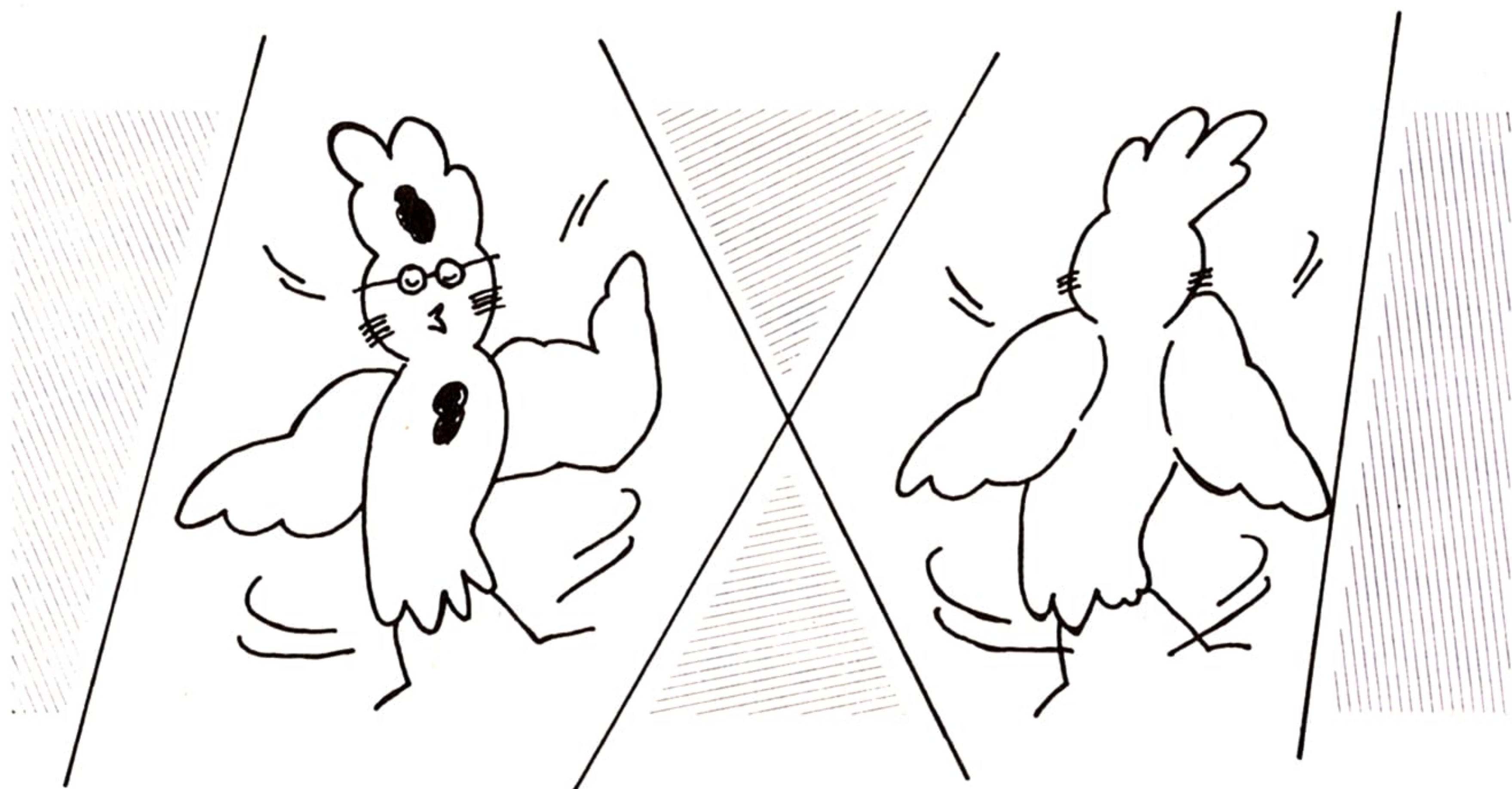
MSX BASIC

4-2 好きな絵をスプライトパターン

自分がデザインした絵を自由自在に動かしてアニメーションを作ることができたら，どんなに楽しいでしょうか．自分のオリジナルなゲームを作って遊ぶことは，ビギナーの夢のひとつです．

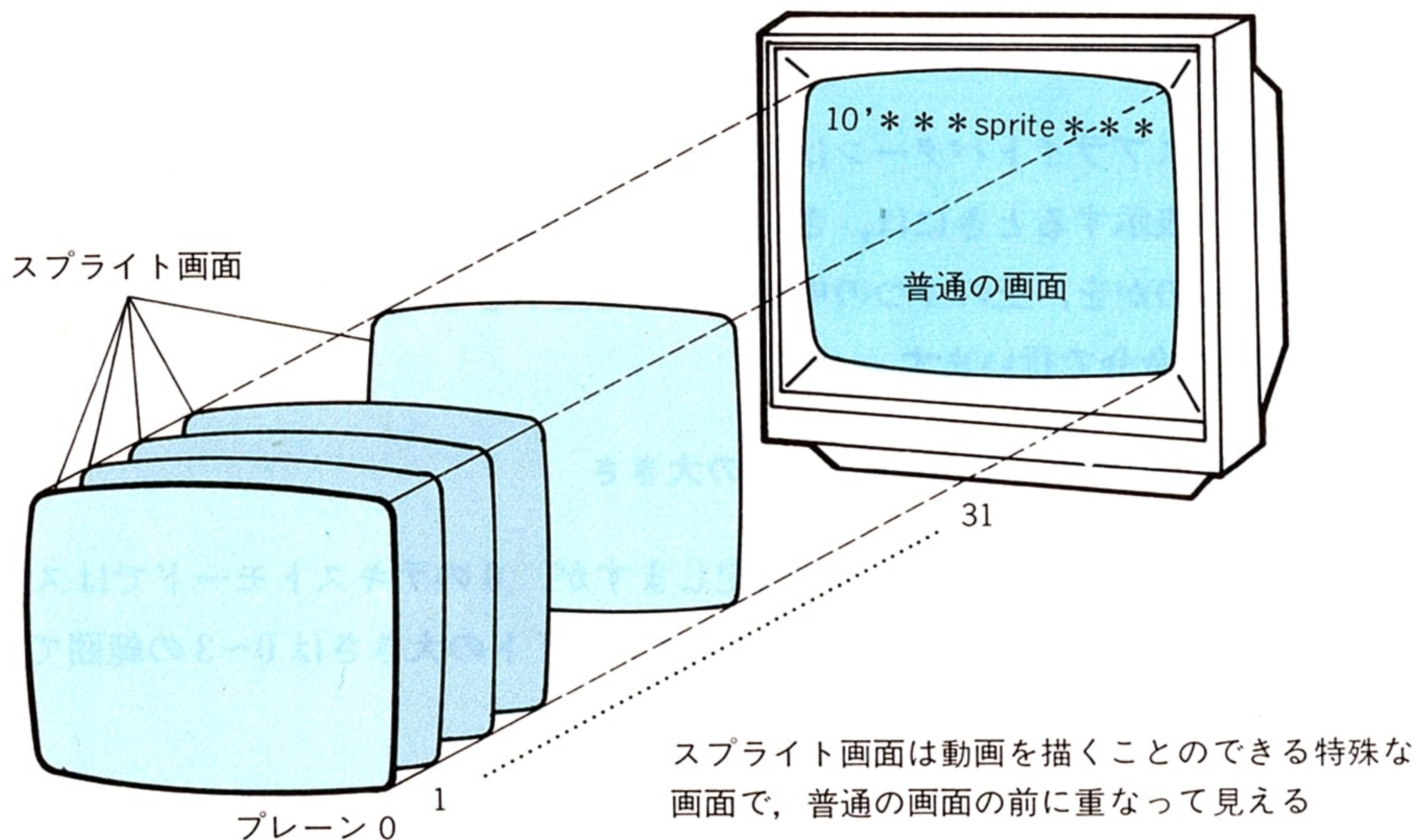
このsectionでは，この夢を簡単にかなえてくれるスプライトの機能を説明していきます．まずスプライトがどのようなものかを説明したあと，実際に絵のパターンを作ってみることにしましょう．

スプライト画面とは

前に説明したとおり，MSXの画面には，グラフィックスと，文字の両方の表示ができ，どちらを選択するかは，SCREEN命令を使って指定します．しかし，MSXには，こうした画面とは別に，スプライト画面と呼ぶ特別な画面が用意されています．このスプ

ライト画面は全部で32面あり，自分で定義した絵のパターン（スプライトパターンと呼びます）を自由に描くことができます．

この1枚1枚のスプライト画面を，プレーンと呼んでいます．

1枚のプレーンには，1つのスプライトパターンを表示することができますから，同時に32個のパターンを表示することができます．ただし，横に並べて表示できるのは，4つまでです．

スプライトパターンは，8×8，または16×16の点を組み合わせて作ります．スプライトパターンは，8×8の小さいパターンならば256個，16×16の大きなパターンならば，64個まで決めておくことができます．

スプライト画面と普通の画面の関係

では，このスプライト画面を使うには，どんな手順を踏めばよいのでしょうか．この手順は，おおよそ次の3段階に分けられます．

① スプライト使用の条件設定

② スプライトパターンの定義

③ スプライトパターンの表示

以下，具体的な手順を説明していきましょう．

スプライト使用の条件設定

スプライトを使うときには，まずスプライトパターンをどの大きさにするかを決める必要があります．

スプライトパターンの大きさには次に示す4つの大きさがあります．左の数字は，このあとに説明するSCREENで指定する数字です．

0・・・8×8　　拡大なし
1・・・8×8　　拡大あり
2・・・16×16　拡大なし
3・・・16×16　拡大あり

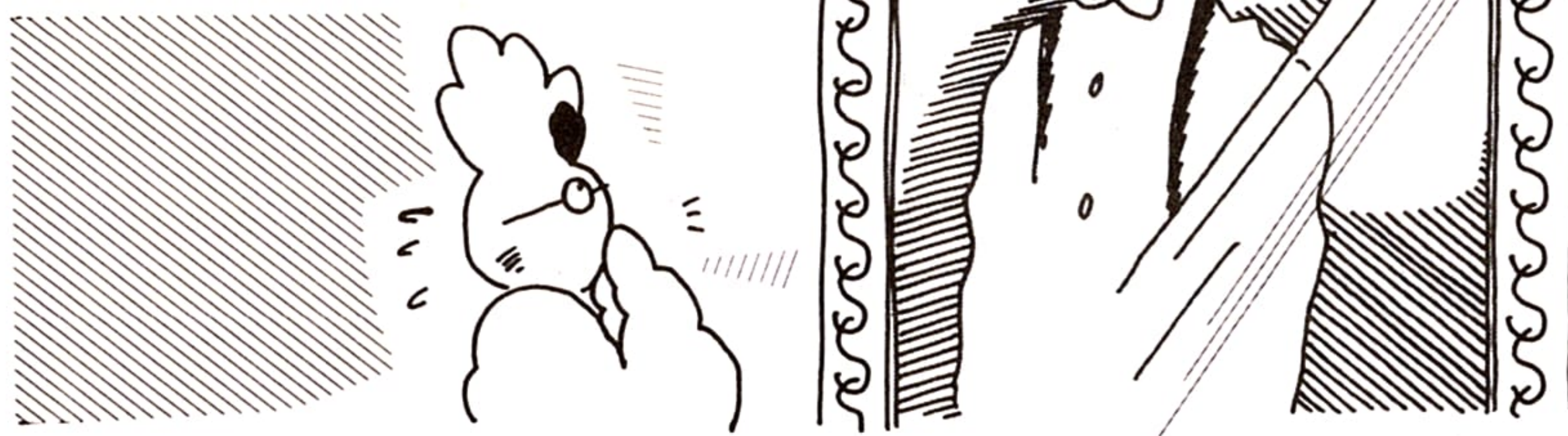
さきほど，スプライトパターンは8×8，または16×16の点を組み合わせて作ると説明しましたが，表示するときには，さらにそれを拡大して表示するのか，そのままの大きさで表示するのかを，上の4つの中から選んで指示しなければなりません．この条件設定もSCREEN命令で行います．

SCREEN　画面モード，スプライトの大きさ

画面モードは前で説明した0〜3で指定しますが，0のテキストモードではスプライトが使えないので，1〜3の間で指定します．スプライトの大きさは0〜3の範囲で，スプライトパターンの大きさを指定します．

たとえば，

```
SCREEN　1 , 1
```

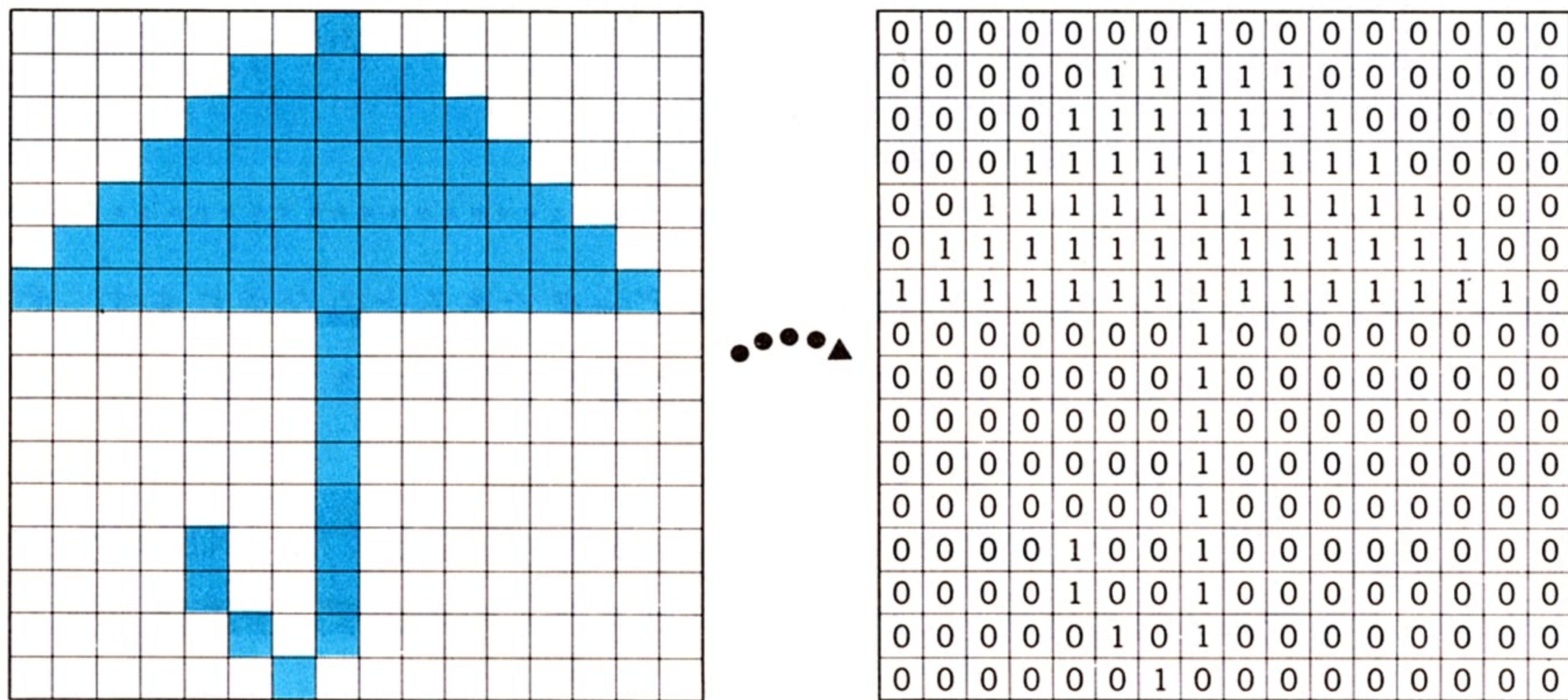
とすると，テキストモードIIで，8×8のパターンを拡大して表示する，という意味になります．

スプライトパターンの定義

スプライト使用の条件設定が済んだらスプライトパターンの定義をします．最初の目標として，傘の絵を16×16の点の組み合わせで定義してみることにしましょう．スプライトパターンの定義は，絵のパターンのデータを用意して，それを加工して行います．

〔1〕イメージパターンの作成

まず16×16のます目を用意して，ます目を塗るか，塗らないかで絵を描いていきます．次に，塗ったところを1，塗らないところを0というように数字に置き換えます．

•••▲

0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	0	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	0	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	0	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0

この0と1の数字が，そのまま絵のパターンのデータとなります．

〔2〕イメージパターンの保存

ここまでの作業は，コンピュータを使わない手作業の部分です．こうして作ったデータを，コンピュータの中に取り込まなければなりません．データを変数に読み込むのにはINPUTがありましたね．しかし，INPUTではプログラムを走らせるたびに，キーボードからデータを打ち込まなくてはなりません．ここではもっと他の方法を考えてみましょう．それは，READとDATAを用いる方法です．DATAは，プログラムの中にひとまとまりのデータを用意しておくときに使います．たとえば，さきほど作ったパターンのデータをプログラムの中に用意するときには，次のようにします．

リスト

```
1000 DATA 0000000100000000
1010 DATA 0000011111000000
1020 DATA 0000111111100000
1030 DATA 0001111111110000
1040 DATA 0011111111111000
1050 DATA 0111111111111100
1060 DATA 1111111111111110
1070 DATA 0000000100000000
1080 DATA 0000000100000000
1090 DATA 0000000100000000
1100 DATA 0000000100000000
1110 DATA 0000000100000000
1120 DATA 0000100100000000
1130 DATA 0000100100000000
1140 DATA 0000010100000000
1150 DATA 0000001000000000
```

行の先頭にDATAがあると，その行の文字，数はデータであるとみなされます．データはカンマで区切ることによって，いくつも同じ行に書くことができますが，ここでは1つの行に1つのデータを書いています．

こうして用意したデータを，変数の中に読み込む命令がREADです．INPUTがキーボードからデータを読み込むのに対し，READはプログラムの中のデータを読み込んでいきます．

〔3〕スプライトの定義

このようにプログラムの中に用意したデータを，実際にパターンとして定義するには，次のようにします．

リスト

```
120 FOR I=1 TO 16
130  READ D$
140  A$=A$+CHR$(VAL("&B"+LEFT$(D$,8)))
150  B$=B$+CHR$(VAL("&B"+RIGHT$(D$,8)))
160 NEXT I
170 SPRITE$(0)=A$+B$
```

120～160行でイメージパターンのデータを変数に取り込んでいます．ここで使っているVALという命令は，文字変数を数値に直す働きをします．170行が，実際にスプライトパターンを定義しているところです．

スプライトパターンの定義部分

170 SPRITE$(n)=A$+B$
→ A$に左半分，B$に右半分の圧縮されたスプライトパターンが入っている
↓
スプライトパターンの登録番号　0～63（16×16）
0～255（8×8）

ここでは，さきほどの傘のパターンを，登録番号0で登録（定義）したことになります．

スプライト使用の条件設定，スプライトの定義，スプライトパターンのデータ，の3つの部分のリストをまとめると，下のようになります．次のsectionで，ここで作ったパターンを実際に表示させ，動かしてみることにしましょう．

リスト

```
100 '** sprite **
110 SCREEN 1,2 ……… 16×16のスプライトの指定（拡大なし）
120 FOR I=1 TO 16
130  READ D$
140  A$=A$+CHR$(VAL("&B"+LEFT$(D$,8)))
150  B$=B$+CHR$(VAL("&B"+RIGHT$(D$,8)))
160 NEXT I
170 SPRITE$(0)=A$+B$
1000 DATA 0000000100000000
1010 DATA 0000011111000000
1020 DATA 0000111111100000
1030 DATA 0001111111110000
1040 DATA 0011111111111000
1050 DATA 0111111111111100
1060 DATA 1111111111111110
1070 DATA 0000000100000000
1080 DATA 0000000100000000
1090 DATA 0000000100000000
1100 DATA 0000000100000000
1110 DATA 0000000100000000
1120 DATA 0000100100000000
1130 DATA 0000100100000000
1140 DATA 0000010100000000
1150 DATA 0000001000000000
```

（120～170行：スプライトの定義）
（1000～1150行：パターンのデータ）

4-3 コンピュータアニメーション

MSX BASIC

スプライトパターンの登録ができたら，いよいよこれを表示して動かすことにします．スプライトパターンをスプライト画面に描く命令は，PUT SPRITEです．

PUT SPRITE プレーン番号，(横，縦)，色，パターン番号

PUT SPRITEでは，どのプレーンに表示するか，スプライト画面のどこに表示するか，何色で表示するか，どのパターンを表示するか，を上のような型で指定します．

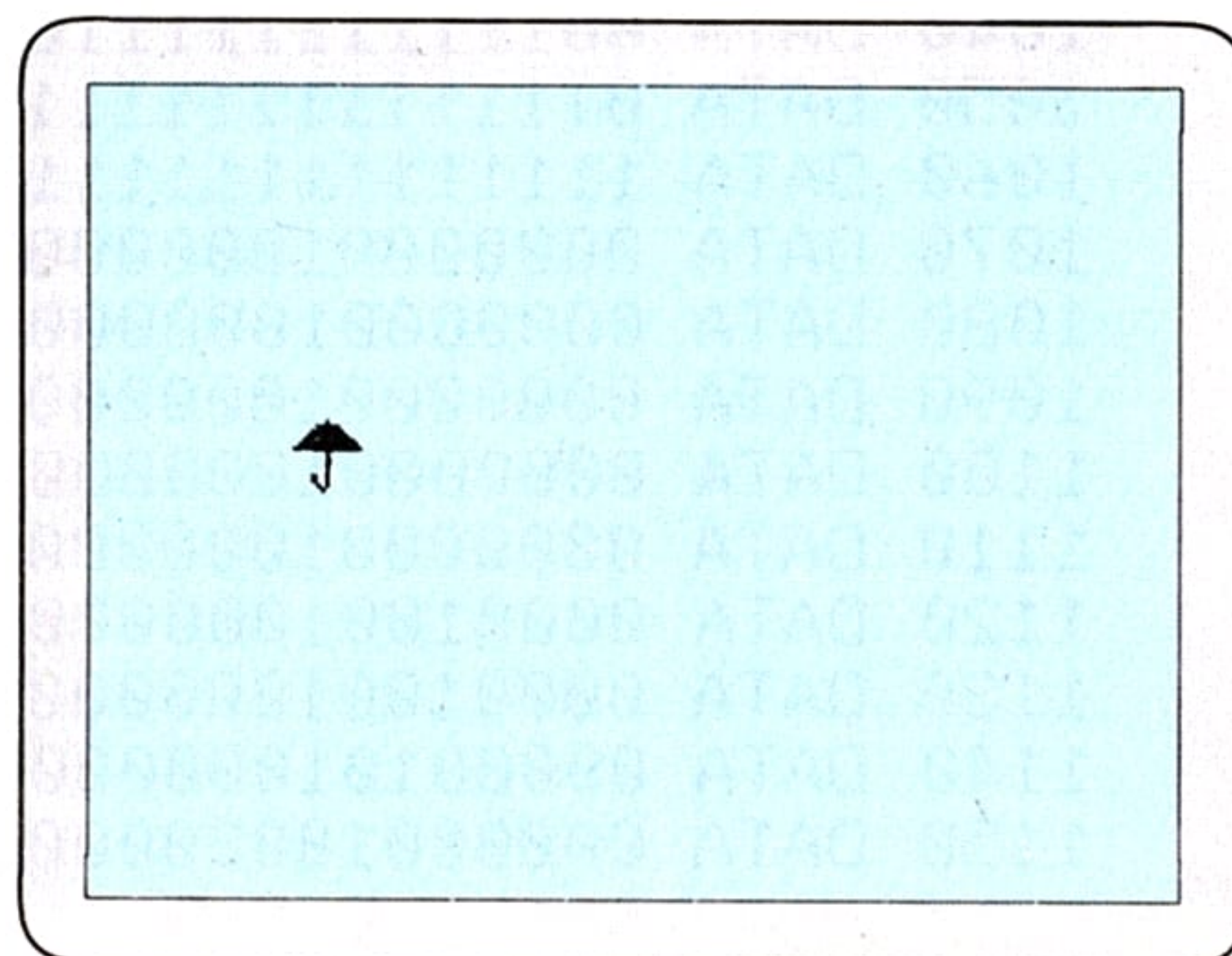
まずさきほどの傘を，32枚あるスプライト画面の一番手前のプレーン0の横50，縦50の位置に白色で表示してみましょう．白のカラーコードは15，傘を登録したプレーンの番号は0ですから，プログラムは次のようになります．

リスト

```
180 CLS
190 PUT SPRITE0,(50,50),15,0
```

180行で画面を消してから，190行でパターンを表示しています．

この2行を4-2のプログラムに追加して実行してみてください．実行する時には F・5 のキーを利用すると便利でしたね．

この結果は右のようになります．

次は傘を１つだけではなく，3つ，色を変えて表示させてみましょう．前に説明したとおり，１枚のプレーンには１つのパターンしか表示できません．たくさんの絵を描こうとすると，何枚もプレーンを使うことになります．そのために，行を追加してみましょう．

プログラムの追加をしたり修正をしたりするときには，画面を消した方が作業がしやすいので，SHIFT ＋ HOME で画面を消しておきます．しかし，どうしたのでしょう．傘は消える気配がありません．実は，SHIFT ＋ HOME とか CLS は，スプライト画面には通用しないのです．スプライトパターンを消すためには，次のようにします．

スプライトパターンの消し方

◦特定のプレーンのパターンを消すとき

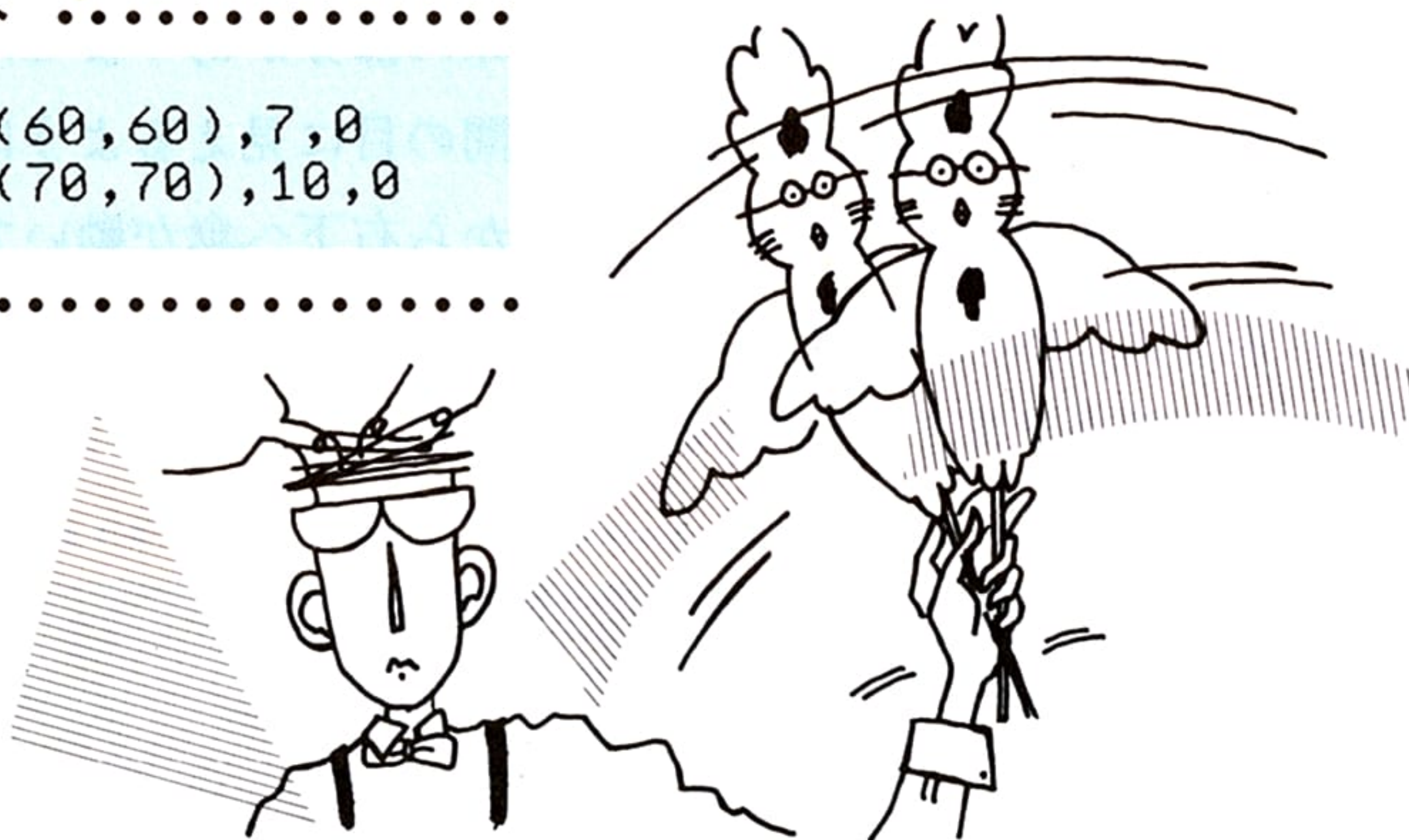
PUT SPRITE プレーン番号，(0，209)

◦スプライト画面全部のパターンを消すとき

PUT SPRITE 0，(0，208)

キーボードから，どちらかの命令を打ち込んで傘を消してください．傘が消えたら200〜210行を加えます．

リスト

```
200 PUT SPRITE1,(60,60),7,0
210 PUT SPRITE2,(70,70),10,0
```

プレーン１の横60，縦60の位置に，水色で１つ，プレーン２の横70，縦70の位置に黄色で１つ，190行で表示したものと合わせて，３つの傘が表示されるはずです．プログラムがうまく動いたら，プレーンや色を変えてみてください．

スプライトパターンを動かす

さあ，いよいよスプライトパターンを動かしてみましょう．スプライトの位置などを変えてみた人は，もう気が付いたかもしれませんが，スプライトには1枚のプレーンに，常に1つのパターンしか表示されないという性質があります．この性質を利用するとスプライトパターンを動かすことができます．PUT SPRITEを使い，横，縦の位置を連続的にずらして表示し続けると，パターンが動いているように見えるのです．

さきほどのプログラムのスプライトを表示している部分を手直しして，プレーン0の上に，表示する位置をずらしながらパターンを描くようにしてみましょう．

リスト

```
180 CLS
190 PUT SPRITE0,(50,50),15,0
195 FOR J=1 TO 100:NEXT J
200 PUT SPRITE0,(60,60),15,0
205 FOR J=1 TO 100:NEXT J
210 PUT SPRITE0,(70,70),15,0
```

195行，205行は，マルチステートメントといって，コロン[：]で区切ることで1つの行に2つ以上の命令が書いてあるのです．ここで，FOR～NEXTを使って繰り返しをしているのですが，見ると繰り返しの中味の部分がありません．これは，パターンの動く速さが速いので，時間かせぎをして人間の目に見えるようにしているのです．

プログラムを実行してみると，左上から右下へ傘が動いていくのがわかります．

次のように，FOR～NEXTで位置を変化させれば，より短いプログラムで連続的に動かすことができます．195行と205行を削除してから実行してみてください．

リスト

```
180 CLS
190 FOR X=10 TO 180
200 PUT SPRITE0,(X,X),15,0
210 NEXT X
```

スプライトパターンを動かす基本は，もうわかりましたね．FOR～NEXTを使わないで，乱数であちこちに動かすこともできるでしょう．

カーソル移動キーで動かす

スプライトパターンの動かし方の基本がわかったところで，もっと自分の意のままにパターンを動かしてみましょう．キーボードの右の方には，カーソルを動かすためのキー(カーソル移動キー)があります．

普通カーソル移動キーを押すと，カーソルは矢印の方向に上下左右に動き，となりあった2つのキーを同時に押すと斜めに動きます．ここではスプライトで定義した傘のパターンを，このカーソル移動キーに合わせて動かしてみることにします．

どのカーソル移動キーが押されているかを調べる命令として，STICKがあります．STICKはもともと，ゲームなどに使うジョイスティックがどの方向に倒されているかを調べるためのものですが，カーソル移動キーにも使うことができます．

STICKは，次のような型で使い，ジョイスティック，またはカーソルの押されている方向で0～8の値をとります．

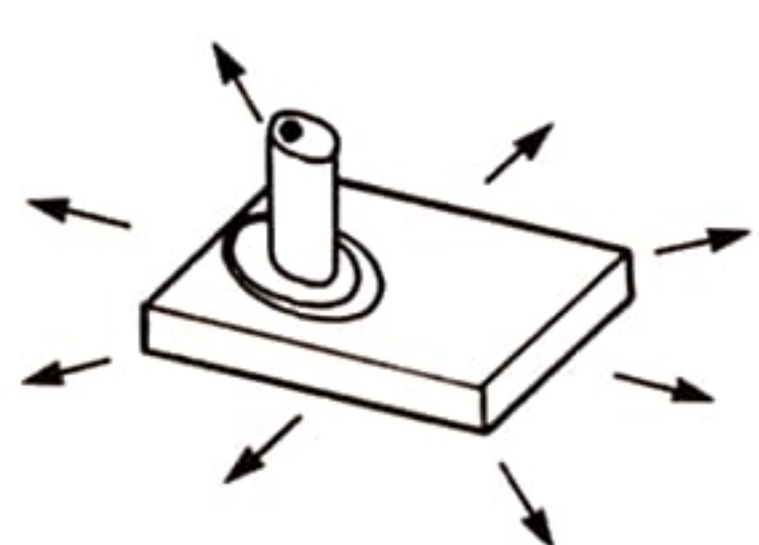
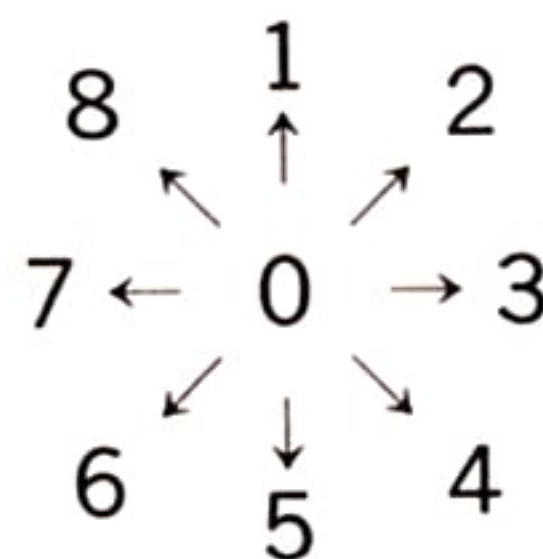
STICKの使い方

```
STICK（n）  n：0→カーソルキー
               1→ジョイスティック1
               2→ジョイスティック2
```

（例） A＝STICK（1）：ジョイスティック1の方向に対応する値を変数Aに入れる

〈ジョイスティックの方向とSTICKの値〉

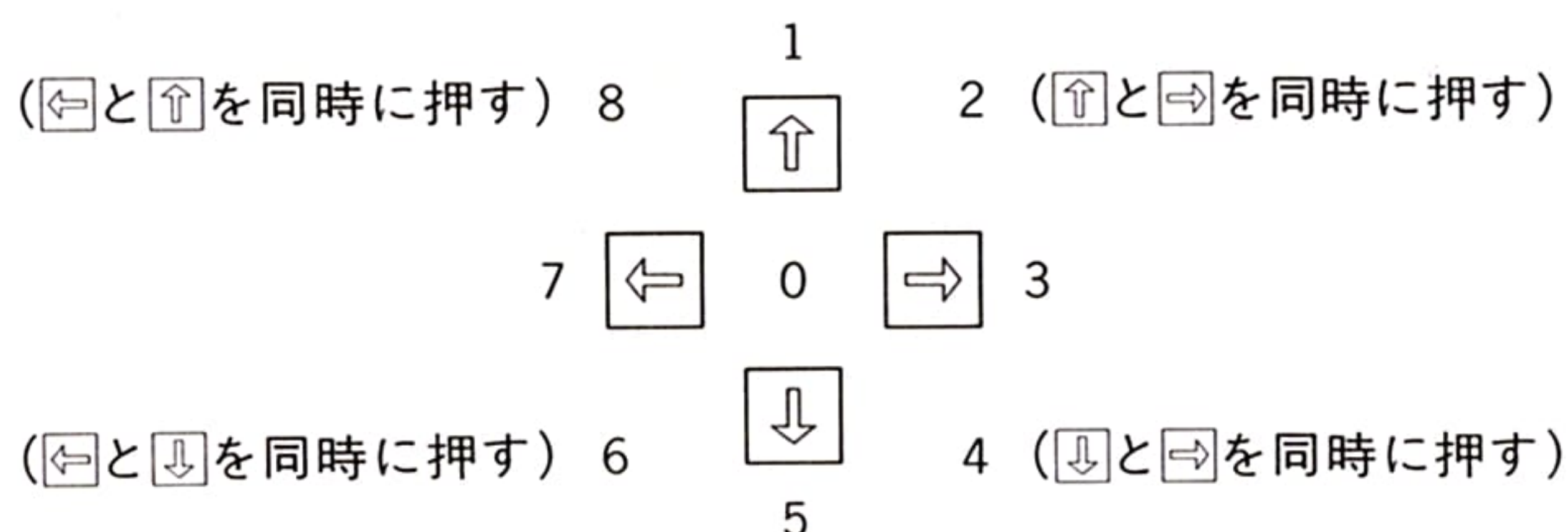
8つの方向に倒せる

```
8  1  2
 ↖ ↑ ↗
7← 0 →3
 ↙ ↓ ↘
6  5  4
```

★スティックを倒した方向によりSTICK(n)の値が決まる
★どの方向にも倒されていないときの値は0

〈カーソルキーの押し方とSTICKの値〉

（⇦と⇧を同時に押す）8　　1 ⇧　　2（⇧と⇨を同時に押す）

7 ⇦　　0　　⇨ 3

（⇦と⇩を同時に押す）6　　⇩ 5　　4（⇩と⇨を同時に押す）

★となりあった2つのキーを同時に押すとななめの方向を示す
★どのキーも押されていないときの値は0

カーソル移動キーを使うときには，STICK(0)の値をチェックして，1ならば上にいく，2ならば斜め右上に，3ならば右に……というように，座標の横，縦を変化させていけばよいのです．これを今までの知識で実現すると，1～8の状態に合わせて8つのIF～THENの組みが必要になります．このようなとき便利なのがON～GOTOです．

```
ON n GOTO a,b,c,...
```

ON～GOTOは上のような型で使います．この命令を使うと，nの値が1ならば行番号a，2ならば行番号b，3ならば行番号c……というように，次に実行する行を変えることができます．

それではこれを使ってプログラムを修正してみましょう．

まず，PUT SPRITEの座標を変数で指定します．次にSTICK(0)を使って，カーソルキーのどれが押されているかをチェックし，その値によって座標を表す変数を増減します．

リスト

```
180 CLS
190 PUT SPRITE0,(X,Y),15,0
```

パターンを動かす部分は，たとえばカーソルキーの[↑]が押されていたら，傘は上に動くのですから，Yの値を減らせばよいということになります．

ON～GOTO，STICK(0)を使って，この部分をプログラムにすると次のようになります．

リスト

```
200 ON STICK(0) GOTO 220,230,240,250,260,270,280,290
210 GOTO 200
220 Y=Y-1:GOTO 190
230 Y=Y-1:X=X+1:GOTO 190
240 X=X+1:GOTO 190
250 X=X+1:Y=Y+1:GOTO 190
260 Y=Y+1:GOTO 190
270 Y=Y+1:X=X-1:GOTO 190
280 X=X-1:GOTO 190
290 X=X-1:Y=Y-1:GOTO 190
```

さあ，実行してみましょう．カーソル移動キーを上下左右斜めに動かすと傘もその動きに応じて動くようになります．ジョイスティックを持っている人は，190行を，STICK(1)としてみてください．

これでスプライトの基本は終わりです．いよいよ次のsectionでは，スプライトを使ってゲームを作ってみましょう．

リスト

```
100 '** sprite **
110 SCREEN 1,2
120 FOR I=1 TO 16
130  READ D$
140  A$=A$+CHR$(VAL("&B"+LEFT$(D$,8)))
150  B$=B$+CHR$(VAL("&B"+RIGHT$(D$,8)))
160 NEXT I
170 SPRITE$(0)=A$+B$
180 CLS
190 PUT SPRITE0,(X,Y),15,0
200 ON STICK(0) GOTO 220,230,240,250,260,270,280,290
210 GOTO 200
220 Y=Y-1:GOTO 190
230 Y=Y-1:X=X+1:GOTO 190
240 X=X+1:GOTO 190
250 X=X+1:Y=Y+1:GOTO 190
260 Y=Y+1:GOTO 190
270 Y=Y+1:X=X-1:GOTO 190
280 X=X-1:GOTO 190
290 X=X-1:Y=Y-1:GOTO 190
1000 DATA 0000000100000000
1010 DATA 0000011111000000
1020 DATA 0000111111100000
1030 DATA 0001111111110000
1040 DATA 0011111111111000
1050 DATA 0111111111111100
1060 DATA 1111111111111110
1070 DATA 0000000100000000
1080 DATA 0000000100000000
1090 DATA 0000000100000000
1100 DATA 0000000100000000
1110 DATA 0000000100000000
1120 DATA 0000100100000000
1130 DATA 0000100100000000
1140 DATA 0000010100000000
1150 DATA 0000001000000000
```

MSX BASIC

4-4 スロットマシンをもう一度

スプライトの基本がわかったところで，さっそく応用のゲームを作ってみましょう．

CHAPTER 3 でスロットマシンゲームを作りましたが，これにスプライト機能を付け加え，数字がグルグル回る部分を，スプライトで作ったパターンに置き換えます．

まず，さきほど作ったプログラムの行番号を整理しておきます．行番号を整理する命令は，RENUMでしたね．ここでRENUM RETURN とすると，行番号が10行から始まって10行おきに，10, 20, 30……と付いてしまいます．そこで，

```
RENUM  100 , 100
```

としてみてください．こうすると，きちんと100行から10行おきに番号が付け直されます．RENUMの正確な使い方を下にまとめておきます．

```
RENUM  a , b , c
```

a：新しく付け直す行番号の最初の行番号

b：もとのプログラムの，行番号を付け直す部分の最初の行番号

c：何行おきの番号を付けるか

3 - 3 のプログラムにRENUMをかけて，行番号を整理すると，次のようになります．

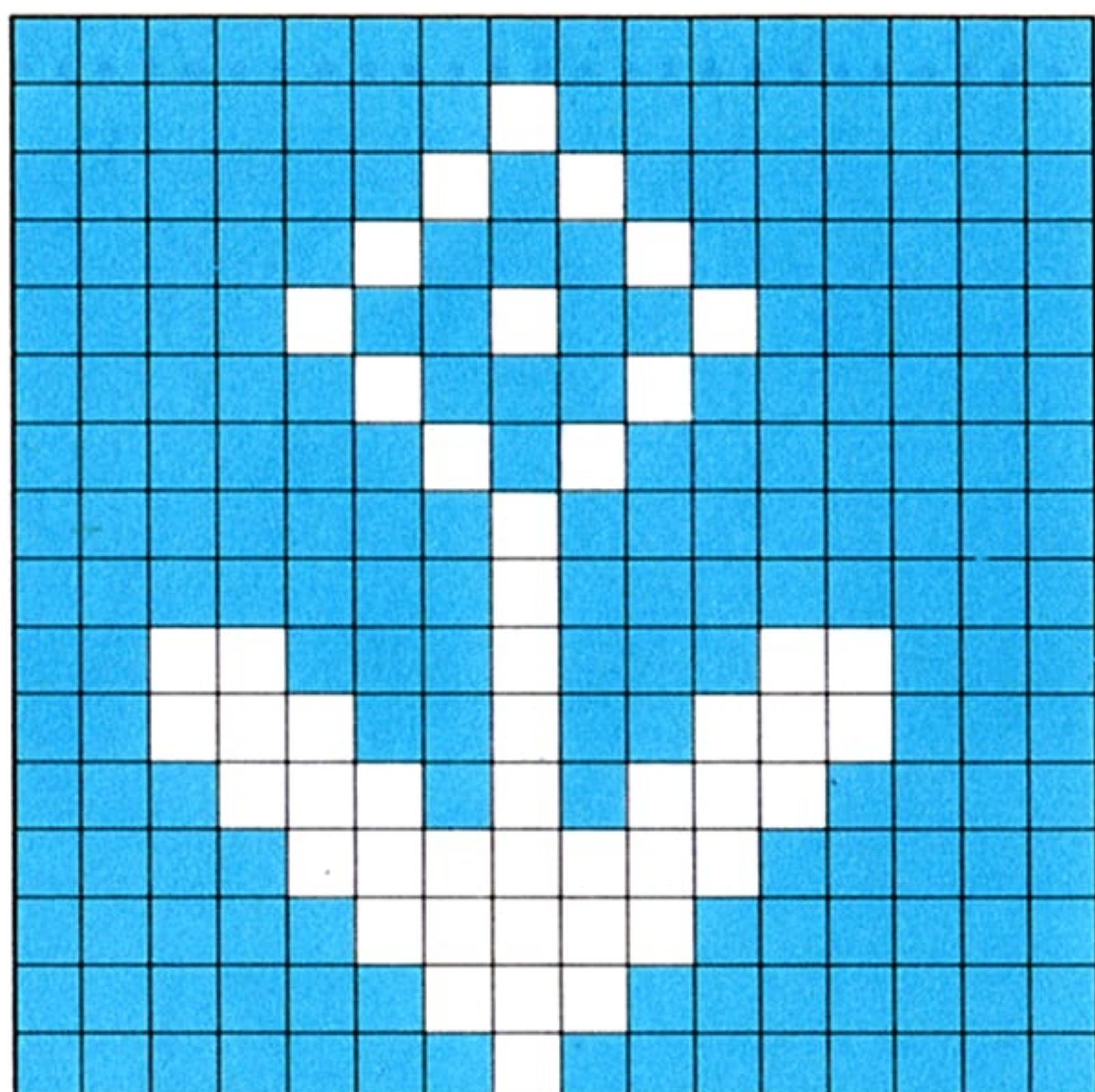
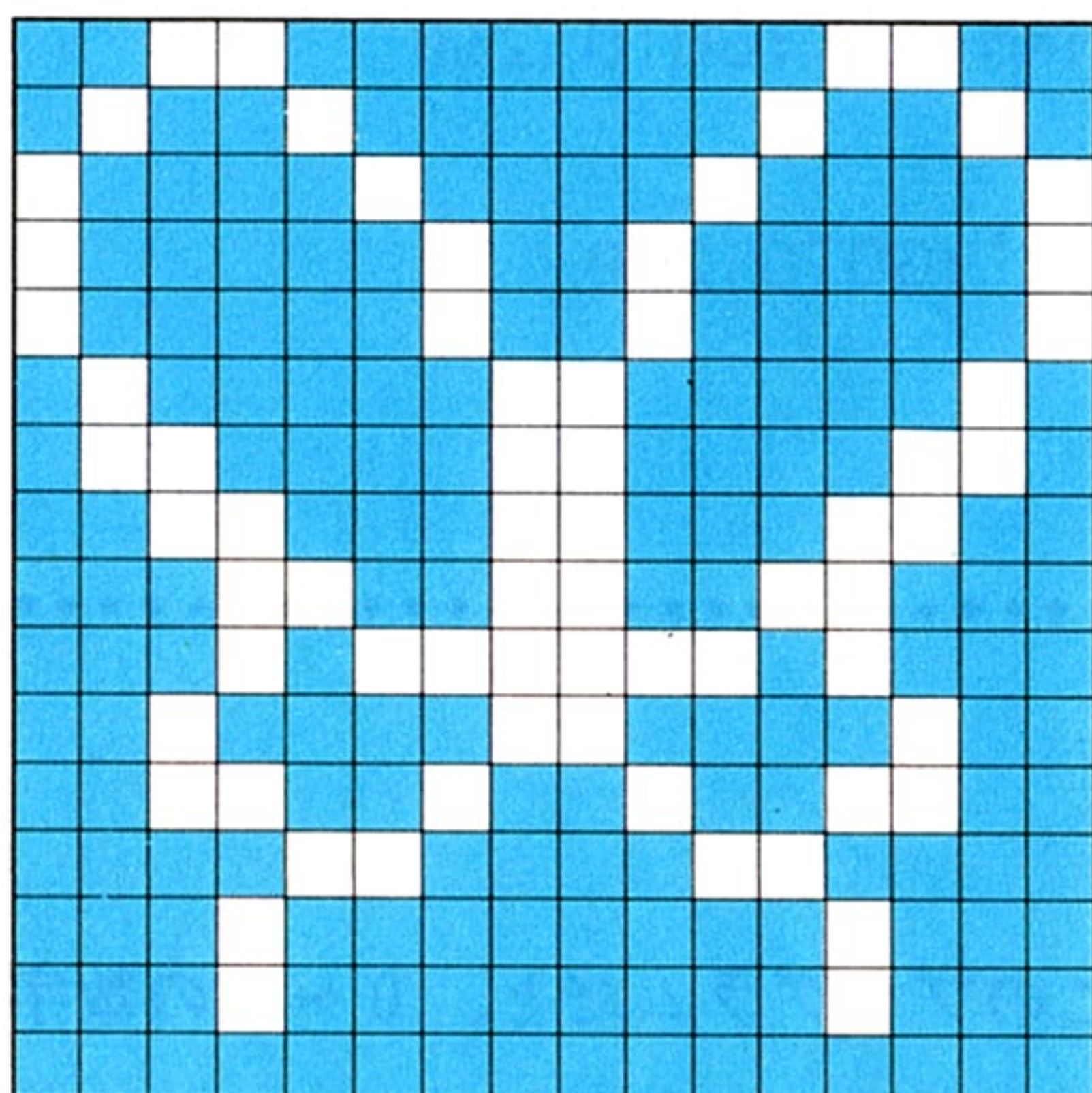
このリストにスプライトの部分を加えていくことにしましょう．

リスト

```
100 'すろっとましん
110 X=RND(-TIME)
120 M=1000
130 PRINT"もちてん=";M
140 INPUT"かけてん=";K
150 IF K>M THEN 130
160 CLS
170 A=INT(RND(1)*10)
180 B=INT(RND(1)*10)
190 C=INT(RND(1)*10)
200 LOCATE 3,5
210 PRINT A;B;C
220 IF INKEY$="" THEN 170
230 IF A=B AND B=C THEN M=M+3*K:GOTO 260
240 IF A=B OR A=C OR B=C THEN M=M+2*K:GOTO 260
250 M=M-K
260 IF M<=0 THEN 290
270 IF M>=10000 THEN PRINT"こうさん!":GOTO 290
280 GOTO 130
290 PRINT"おわり"
300 END
```

スプライトのデータ圧縮

まず，スプライトパターンを作りましょう．このプログラムだと，０～９の数字に対応する10個のデザインを作る必要があります．プログラムを変更してパターンの数を減らし，その代わり各パターンのデザインに凝ってみましょう．

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	1
1	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1
1	1	1	1	1	1	0	0	0	0	0	1	1	1	1	1
1	1	1	1	1	1	0	0	0	0	0	0	1	1	1	1
1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	1
1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1
1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	1
1	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
1	1	1	1	1	1	0	0	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
1	1	1	1	1	1	0	1	0	1	1	1	1	1	1	1
1	1	1	1	1	0	1	1	1	0	1	1	1	1	1	1
1	1	1	1	0	1	1	0	1	1	0	1	1	1	1	1
1	1	1	1	1	0	1	1	1	0	1	1	1	1	1	1
1	1	1	1	1	1	0	1	0	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1
1	1	0	0	1	1	1	0	1	1	1	0	0	1	1	1
1	1	0	0	0	1	1	0	1	1	0	0	0	1	1	1
1	1	1	0	0	0	1	0	1	0	0	0	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0	1	1	1	1	1
1	1	1	1	1	0	0	0	0	0	1	1	1	1	1	1
1	1	1	1	1	1	0	0	0	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1

1	1	0	0	1	1	1	1	1	1	1	1	0	0	1	1
1	0	1	1	0	1	1	1	1	1	1	0	1	1	0	1
0	1	1	1	1	0	1	1	1	1	0	1	1	1	1	0
0	1	1	1	1	1	0	1	1	0	1	1	1	1	1	0
0	1	1	1	1	1	0	1	1	0	1	1	1	1	1	0
1	0	1	1	1	1	1	0	0	1	1	1	1	1	0	1
1	0	0	1	1	1	1	0	0	1	1	1	1	0	0	1
1	1	0	0	1	1	1	0	0	1	1	1	0	0	1	1
1	1	1	0	0	1	1	0	0	1	1	0	0	1	1	1
1	1	1	0	1	0	0	0	0	0	0	1	0	1	1	1
1	1	0	1	1	1	1	0	0	1	1	1	1	0	1	1
1	1	0	0	1	1	0	1	1	0	1	1	0	0	1	1
1	1	1	1	0	0	1	1	1	1	0	0	1	1	1	1
1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1
1	1	1	0	1	1	1	1	1	1	1	1	0	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0	0	1	1	1	1	0	0	1
1	1	1	1	1	0	0	0	0	0	0	1	0	1	1	0
0	0	1	1	0	0	0	0	0	0	0	0	1	1	1	0
1	0	0	0	0	0	0	0	0	0	0	0	1	1	1	0
1	1	0	0	0	0	0	0	0	0	0	0	1	1	1	0
1	1	1	0	0	0	0	0	0	0	0	0	1	1	0	1
1	1	1	1	0	0	0	0	0	0	0	0	0	0	1	1
1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1
1	1	1	1	1	1	0	0	0	0	1	1	1	1	1	1
1	1	1	1	1	0	0	0	0	0	0	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

スプライトパターンを使ったスロットマシンで使用するデザイン

また，さきほどのようにパターンのデータを0と１のイメージそのままのデータの形で持っていると，プログラムの大きさは膨大になってしまいます．そこで，なんとかデータを圧縮することを考えて作ったのが下のプログラムです．

リスト

```
100 '
110 DIM D(32)
120 FOR I=1 TO 16
130  INPUT A$
140  C$="&B"+LEFT$(A$,8)
150  D(I)=VAL(C$)
160  C$="&B"+RIGHT$(A$,8)
170  D(I+16)=VAL(C$)
180 NEXT I
190 '
200 FOR I=1 TO 32
210  PRINT D(I),
220 NEXT I
```

このプログラムに１行分の１と０のパターンのデータを入力すると，それを0〜255の間の数字２つに変換してくれます．たとえば，１行ずつ1100111111110011のようにデータを入力して，16行全部のデータを入力すると，画面に32個の数字が表示されるので，その数字をメモしておくのです．この操作で１つのパターンのデータが32個の数字に圧縮されます．これをプログラムの中に，DATAとしてとっておきましょう．4つのパターンについてこの処理を行うと，データの部分は次のようになります．

リスト

```
1000  '== sprite data ==
1010 DATA 255,255,253,253,252,252,252,252,252,253,192
1020 DATA 192,240,252,255,255,255,255,255,255,255,127
1030 DATA 63,31,15,255,3,7,15,255,255,255
1040 DATA 255,254,253,251,246,251,253,254,254,206,198
1050 DATA 226,240,248,252,254,255,255,127,191,223,191
1060 DATA 127,255,255,231,199,143,31,63,127,255
1070 DATA 207,183,123,125,125,190,158,206,230,232,222
1080 DATA 205,243,239,239,255,243,237,222,190,190,125
1090 DATA 121,115,103,23,123,179,207,247,247,255
1100 DATA 255,255,255,255,254,248,48,128,192,224,240
1110 DATA 248,252,248,255,255,255,255,255,255,121,22
1120 DATA 14,14,14,13,3,31,63,31,255,255
```

また，パターンの数を１０個から4個に減らしたことにより，右のような修正が必要となります．

リスト

```
170 A=INT(RND(1)*4)
180 B=INT(RND(1)*4)
190 C=INT(RND(1)*4)
```

サブルーチンを使う

このプログラムに，①スプライトの条件設定，②スプライトパターンの定義，③スプライトパターンの表示，の３つの部分を付け加えなければなりません．このうち①②の部分はひとまとめにしておいた方が，スプライトパターンを変更したりする場合に便利です．このようなときには，これらの部分をサブルーチンと呼ばれる単位でまとめておき，あとでプログラムの追加・修正をするようにすればよいでしょう．

サブルーチンを使うにはGOSUBとRETURNをペアで用います．サブルーチンを呼ぶときにはGOSUBを使い，サブルーチンから戻るときにはRETURNを使うのです．GOSUBはGOTOと形が似ていますが，GOTOは行きっきりなのに対して，GOSUBはRETURNにより戻ってくるところが大きく違います．

サブルーチンは，プログラムの中で何か所も同じような処理を行う場合によく用いられます．サブルーチンをうまく使えば，大きなプログラムを作るときでも，全体を機能別の部分部分に分けて扱えるので，見とおしのよいプログラムを作ることができます．

また，サブルーチンの中で，他のサブルーチンを呼ぶことも可能です．ただし，GOSUBでサブルーチンを呼んだときには，それに対応するRETURNが存在しなければなりません．GOSUBとRETURNは必ずペアで使用するようにしてください．

スロットマシンゲームのプログラムでは125行のGOSUB　310で，310行からのスプラ

GOSUBとRETURNの使い方

```
100 ' すろっとましん
110 X=RND(-TIME)
120 M=1000
125 GOSUB 310
130 PRINT "もちてん=";M
  :
300 END
```

（サブルーチン）

```
310 ' **sprite**
320 SCREEN 1, 2
 :
400 NEXT I
410 RETURN
```

★サブルーチンは必ずRETURNで終る

イト定義のサブルーチンを実行していきます．そして，410行のRETURNまでくると，再びGOSUBの次の命令に戻ります．ここでは130行に戻ってきます．

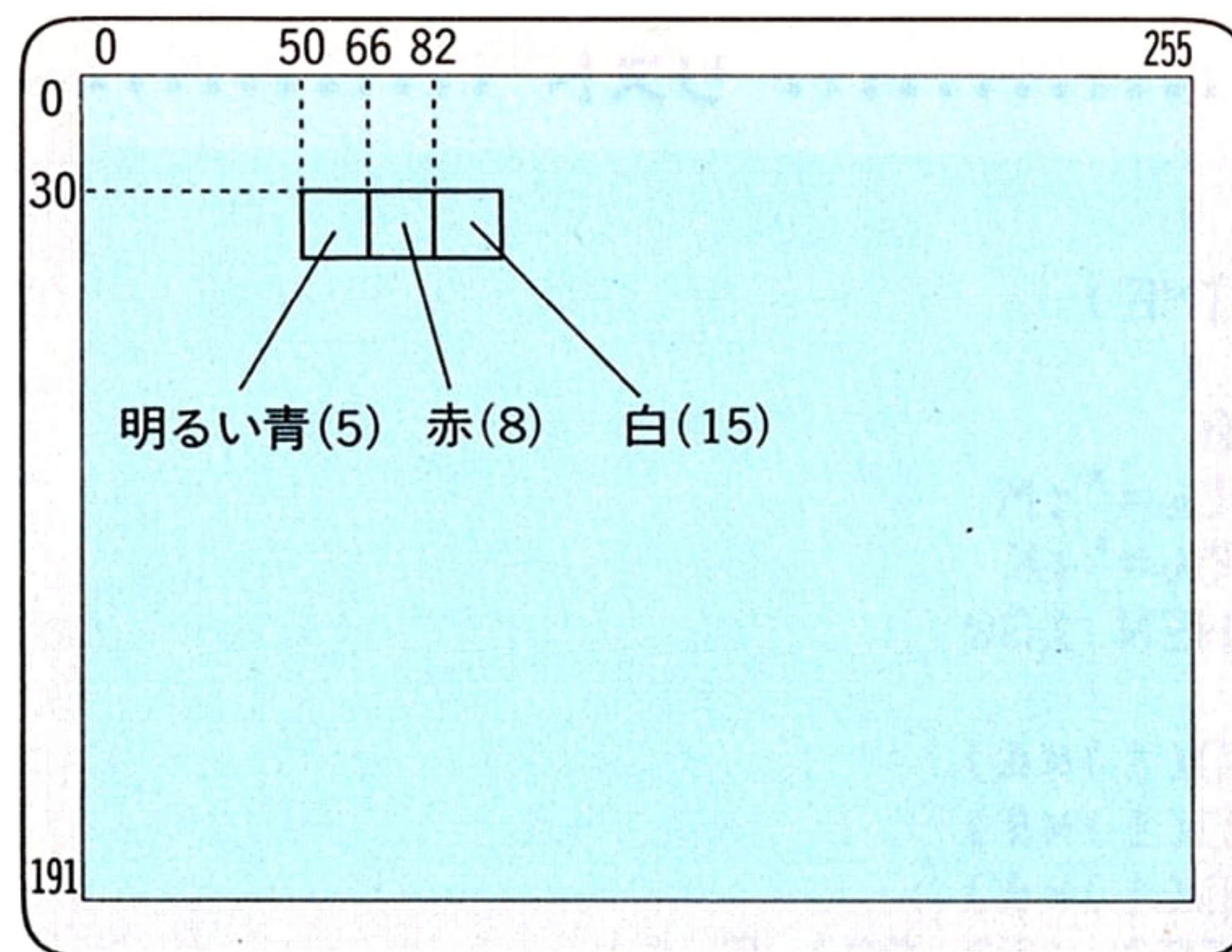
データの持ち方が変わったので，サブルーチンの中のスプライトパターンの登録の部分も変わっています．この部分のリストは右のようになります．

340〜390行の部分は，このように圧縮されたパターンを定義するときのひな形として，覚えておきましょう．330〜400行のFOR〜NEXTで，4つの数字のパターンを，パターン0〜パターン3として登録します．

リスト

```
310 '** sprite **
320 SCREEN 1,2
330 FOR I=0 TO 3
340  SP$=""
350  FOR J=1 TO 32
360   READ D
370   SP$=SP$+CHR$(D)
380  NEXT J
390  SPRITE$(I)=SP$
400 NEXT I
410 RETURN
```

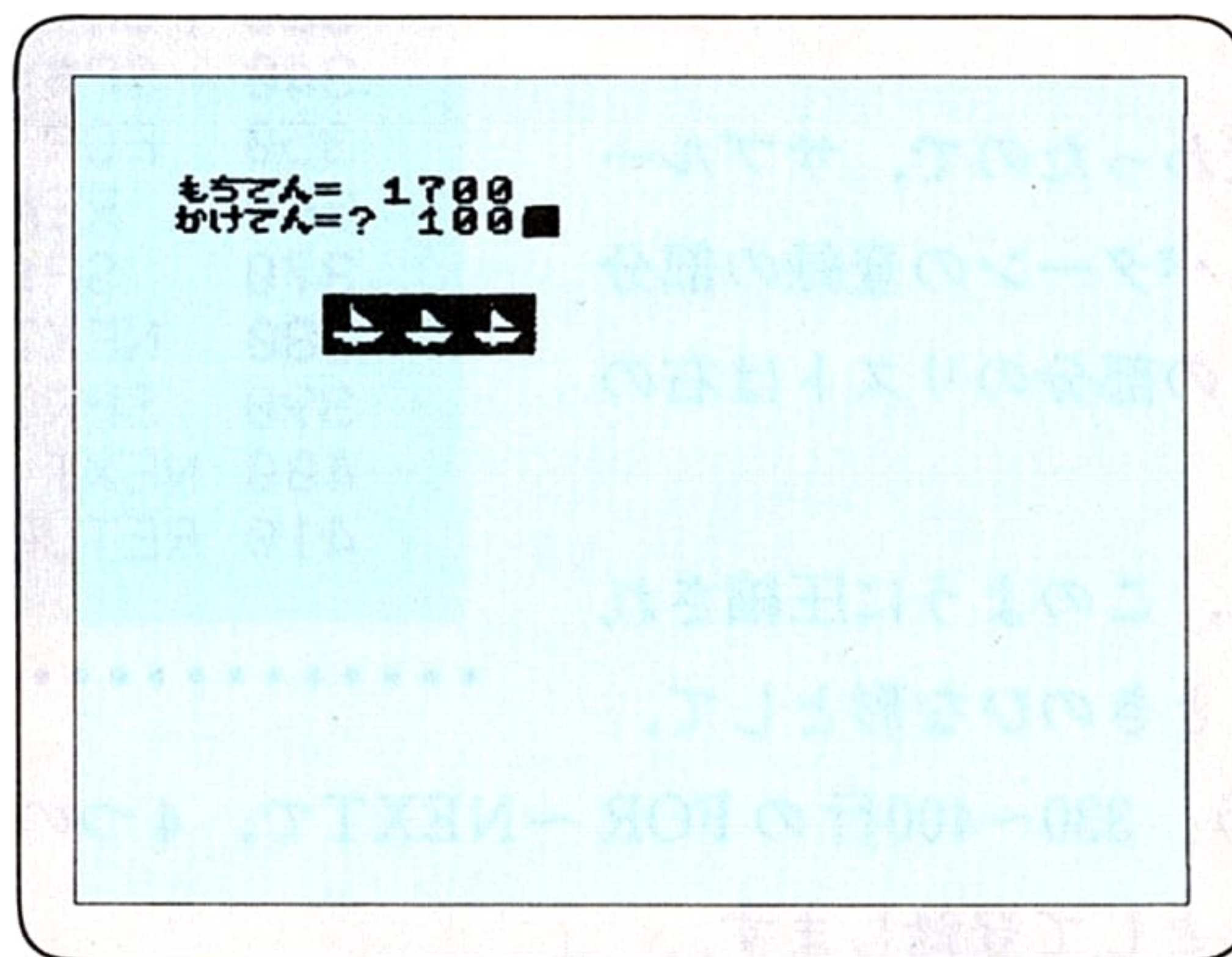
スプライトの表示

スプライトパターンが登録できたら，あとはゲームの中の表示の部分をPUT SPRITEに変えるだけです．

スプライトパターンの表示位置(上)とパターン表示するためのリスト(下)

リスト

```
200 PUT SPRITE0,(50,30),5,A
210 PUT SPRITE1,(66,30),8,B
215 PUT SPRITE2,(82,30),15,C
```

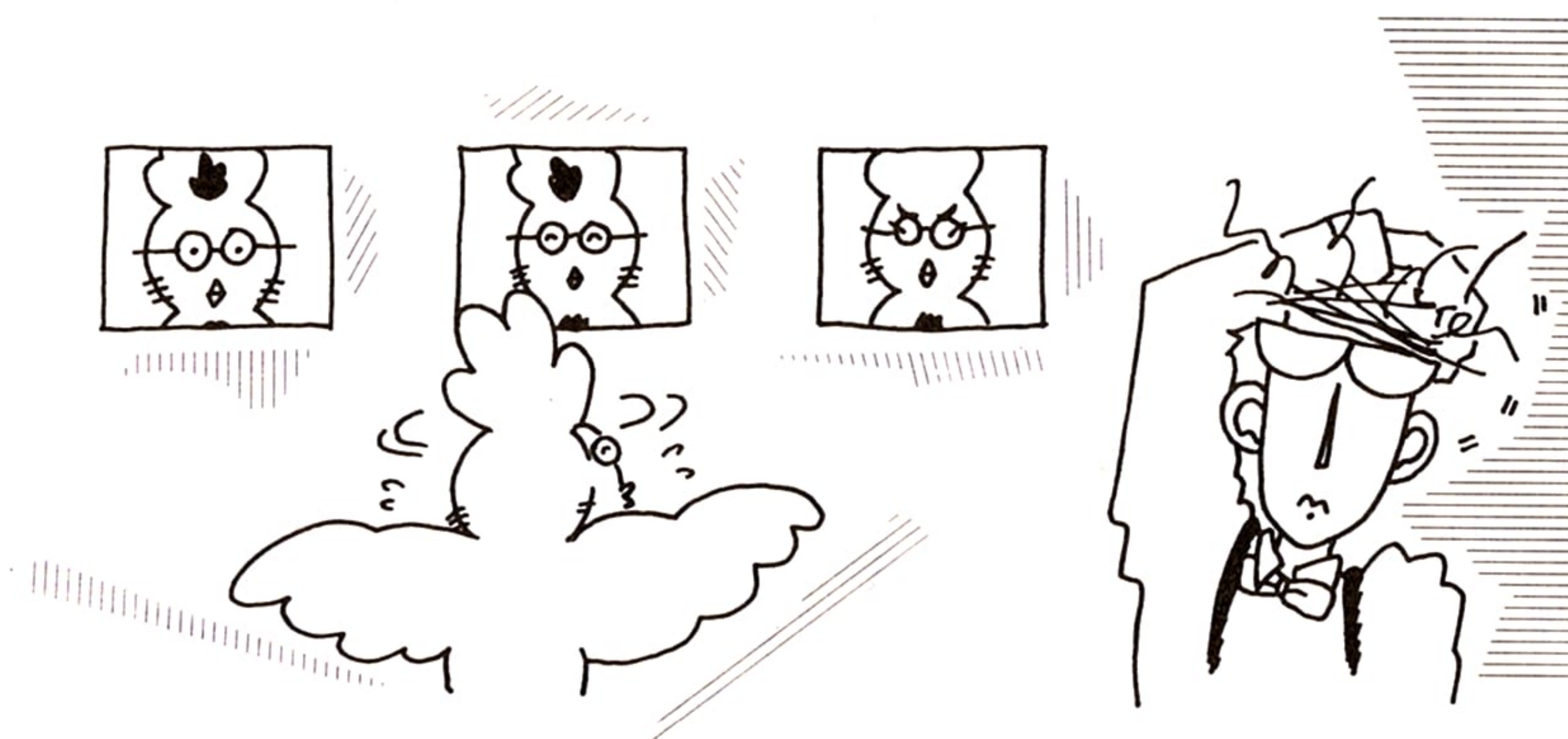
表示にスプライトを使ったことで，ゲームはどう変わったでしょうか．データの打ち込みやプログラムの手直しが終わったら実行してみましょう．

```
もちてん= 1700
かけてん=? 100
```

基本的な構造は変わっていないのに，表示を変えただけで，見違えるようになりました．音を付け加えたりすれば，もっとおもしろくなるでしょう．ここで挙げたものは，ほんの一例にすぎません．あなたも，自分自身でいろいろな工夫を加えてみてください．

リスト

```
100 'するっとましん
110 X=RND(-TIME)
120 M=1000
125 GOSUB 310
130 PRINT"もちてん=";M
140 INPUT"かけてん=";K
150 IF K>M THEN 130
160 CLS
170 A=INT(RND(1)*5)
180 B=INT(RND(1)*5)
190 C=INT(RND(1)*5)
200 PUT SPRITE0,(50,30),5,A
210 PUT SPRITE1,(66,30),8,B
215 PUT SPRITE2,(82,30),15,C
220 IF INKEY$="" THEN 170
230 IF A=B AND B=C THEN M=M+3*K:GOTO 260
240 IF A=B OR A=C OR B=C THEN M=M+2*K:GOTO 260
250 M=M-K
```

```
260 IF M<=0 THEN 290
270 IF M>=10000 THEN PRINT"こうさん!":GOTO 290
280 GOTO 130
290 PRINT"おわり"
300 END
310 '** sprite **
320 SCREEN 1,3
330 FOR I=0 TO 4
340  SP$=""
350  FOR J=1 TO 32
360   READ D
370   SP$=SP$+CHR$(D)
380  NEXT J
390  SPRITE$(I)=SP$
400 NEXT I
410 RETURN
1000 '== sprite data ==
1010 DATA 255,255,253,253,252,252,252,252,252,253,192
1020 DATA 192,240,252,255,255,255,255,255,255,255,127
1030 DATA 63,31,15,255,3,7,15,255,255,255
1040 DATA 255,254,253,251,246,251,253,254,254,206,198
1050 DATA 226,240,248,252,254,255,255,127,191,223,191
1060 DATA 127,255,255,231,199,143,31,63,127,255
1070 DATA 207,183,123,125,125,190,158,206,230,232,222
1080 DATA 205,243,239,239,255,243,237,222,190,190,125
1090 DATA 121,115,103,23,123,179,207,247,247,255
1100 DATA 255,255,255,255,254,248,48,128,192,224,240
1110 DATA 248,252,248,255,255,255,255,255,255,121,22
1120 DATA 14,14,14,13,3,31,63,31,255,255
```

MSX BASIC

4-5 MSXらくがき帳

MSXのグラフィックスとして，4－2から4－4の3つのsectionに渡り，スプライト機能を中心に説明をしてきました．

ここでは，MSXのグラフィックスの第2の特徴であるDRAW命令を使って，画面の上に自由に絵を描いてみることにしましょう．

DRAW命令では画面の上にペンがあるものとして，このペンを上にいくつ，下にいくつ，斜めにいくつ，というように動かして絵を描いていきます．DRAW命令を使うことによって，座標を気にすることなく，あたかもペンを動かすような感覚で絵を描くことができるのです．

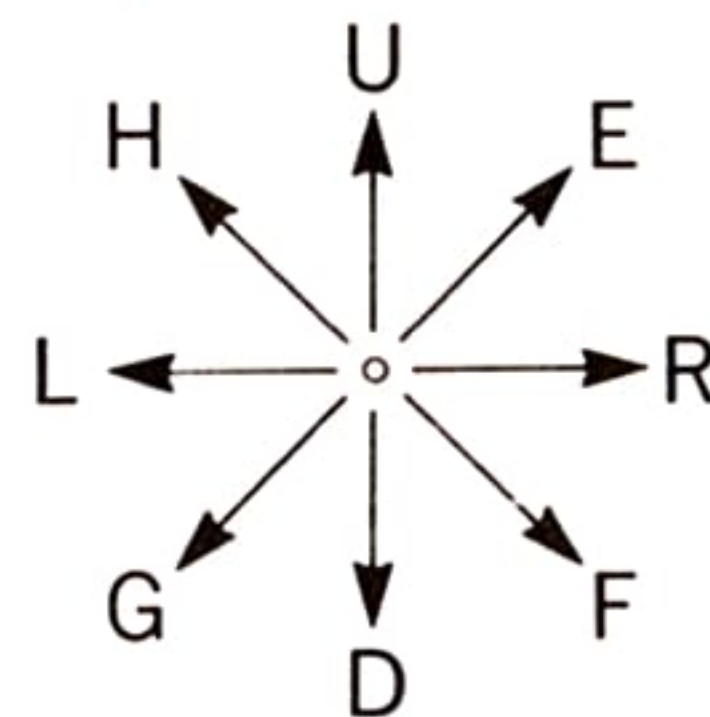
グラフィックマクロ命令

ペンでどのように絵を描くのか，その手順を具体的に表すのがグラフィックマクロ命令です．このグラフィックマクロ命令とDRAWを組み合わせて使えば，線で構成されたさまざまな図形を描くことができます．

DRAWとグラフィックマクロ命令を組み合わせた使い方は，次のとおりです．

DRAW "グラフィックマクロ命令"

グラフィックマクロ命令のうち，代表的なものを右に示しておきます．UとかDとかで移動する方向を，nの部分で移動する長さを表します．

グラフィックマクロ命令

ペンを

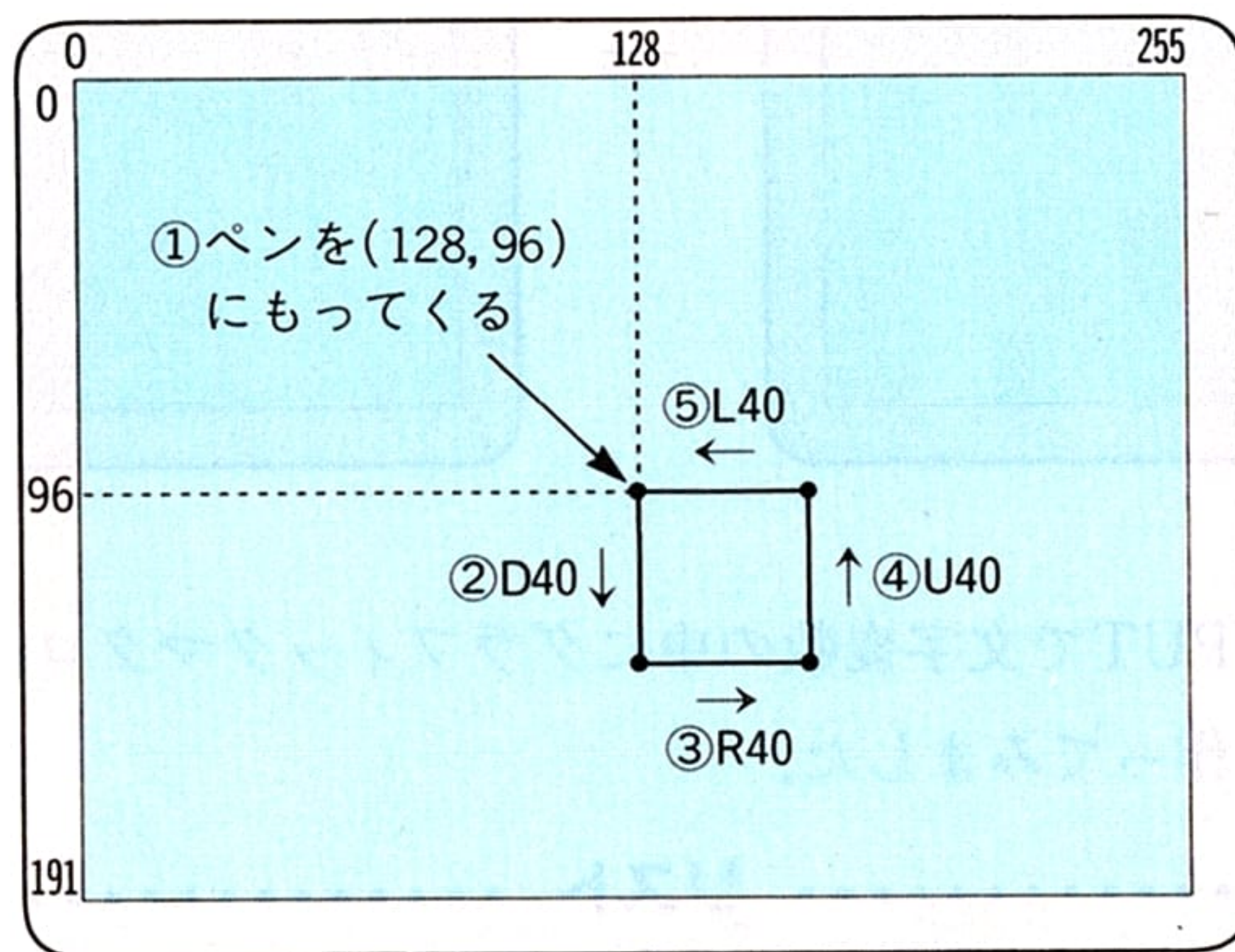
Un	上へ点n個分移動	En	右上へ点n個分移動
Dn	下へ点n個分移動	Fn	右下へ点n個分移動
Ln	左へ点n個分移動	Gn	左下へ点n個分移動
Rn	右へ点n個分移動	Hn	左上へ点n個分移動

★BM x，y　横x，縦yの位置にペンを置く

グラフィックマクロ命令とは，画面の上にペンが下りてきて，そのペンに動きを指示するものだと考えればよいでしょう．たとえばU40とすると，今ペンのあるところから上に40ペンが移動するので，線が引けることになります．理解しやすいように例を挙げておきます．下のリストを見てください．

リスト

```
10 SCREEN 2
20 DRAW"BM128,96"
30 DRAW"D40R40U40L40"
40 GOTO 40
```

DRAW命令は，グラフィック画面でしか使えないので，10行でSCREEN 2を指定します．20行でペンの位置を，横128，縦96の位置に下ろして，30行で絵を描いています．前のページの図と見比べて，どのような絵を描いているのか想像してください．下に40，右に40，上に40，左に40と，つまりは正方形を描いているのです．この様子を，実際にプログラムを動かした画面で確認してみてください．

DRAWの次の" "で囲まれるグラフィックマクロ命令は，大文字でも小文字でもかまいませんが，数字の1と，アルファベットの小文字の l（エル）が判別しにくいので，CAPキーを押してすべて大文字で打ち込んであります．ここでは一番単純に正方形を描いてみましたが，さきに挙げたグラフィックマクロ命令を使いこなすと，もっと複雑な絵も描けるはずです．

キーボードから指示する

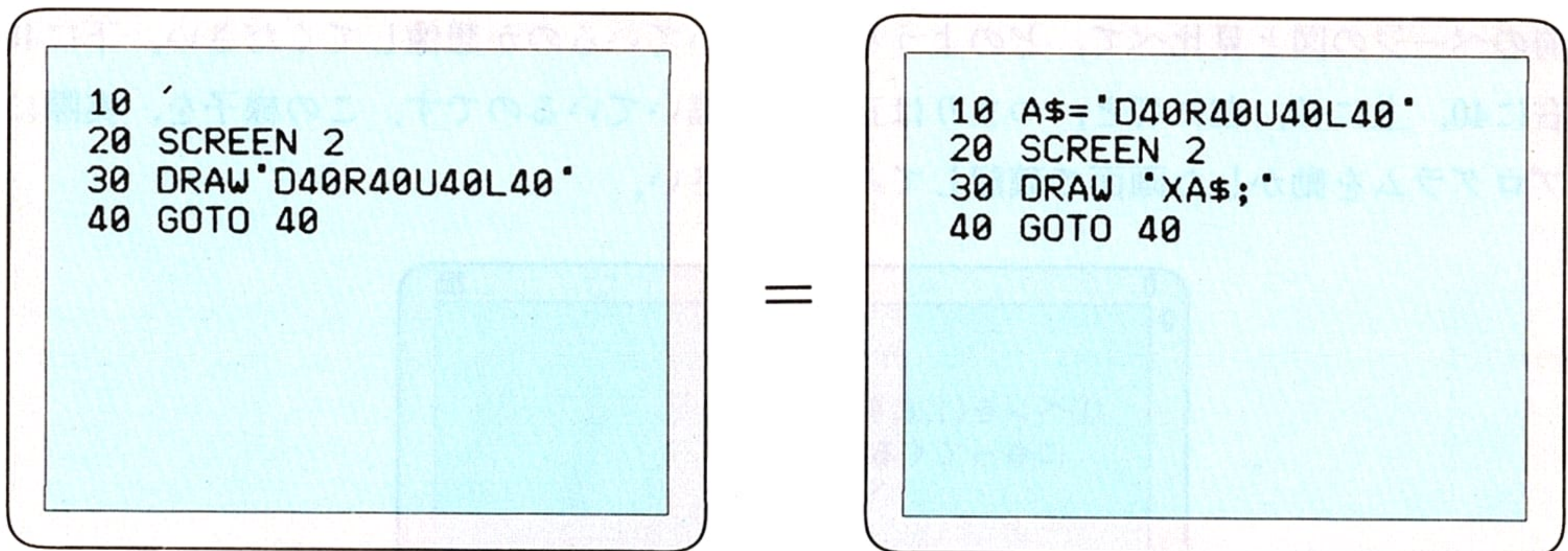
一番単純なDRAWの使い方は，このようにプログラムの中で，描く図形を決めてしまうやり方です．次のステップとして，キーボードからグラフィックマクロ命令を打ち込んで，プログラムを直さなくても，いろいろな図形が描けるようにしてみましょう．グラフィックマクロ命令は，次のような型で，変数を使って指定することができます．

X○$;

○の部分で変数を識別します．これはたとえば，XA$；のように使いますが，文字変数A$の中に，R20F30D60などのグラフィックマクロ命令を入れるのです．ですから，次の2つは全く同じ動きをします．

```
10 '
20 SCREEN 2
30 DRAW"D40R40U40L40"
40 GOTO 40
```

=

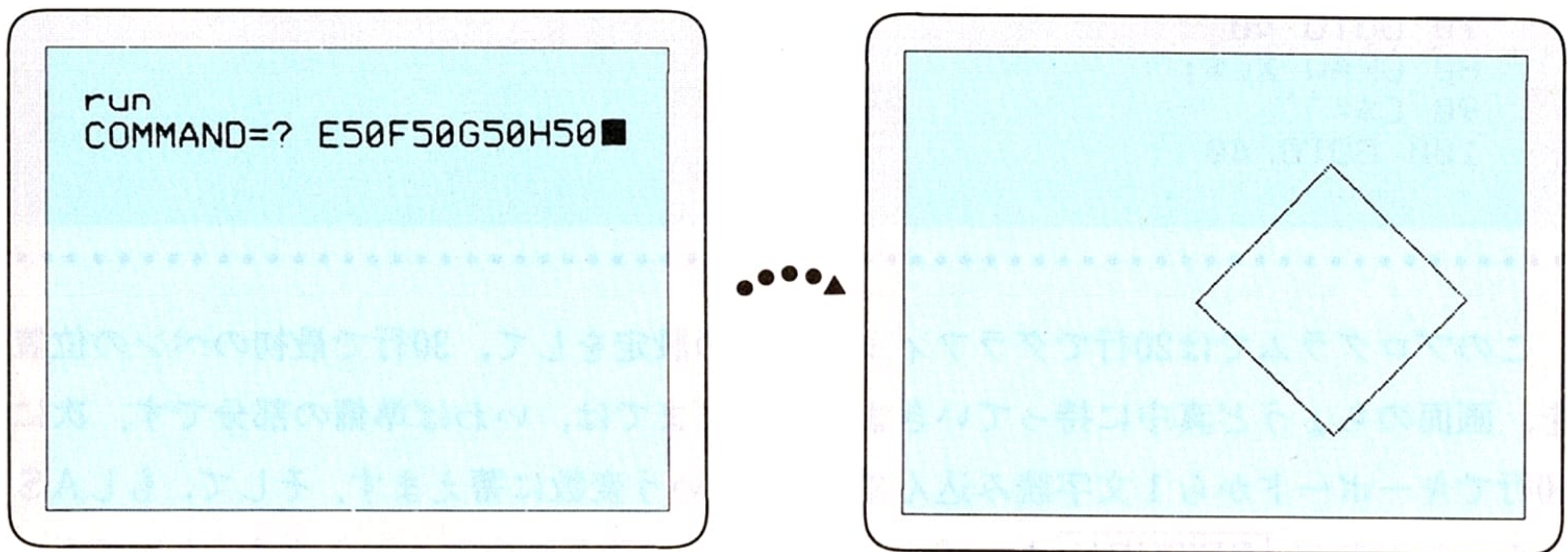
```
10 A$="D40R40U40L40"
20 SCREEN 2
30 DRAW "XA$;"
40 GOTO 40
```

これを利用して，INPUTで文字変数の中にグラフィックマクロ命令を入れ，画面に表示させるプログラムを作ってみました．

リスト

```
10 'DRAW
20 INPUT"COMMAND=";C$
30 SCREEN 2
40 DRAW"BM128,96"
50 DRAW "XC$;"
60 IF INKEY$="" THEN 60
70 GOTO 20
```

COMMAND＝？の問いに，グラフィックマクロ命令を入れてやると，そのとおりの線を引きます．60行で，何かキーが押されているかをチェックして，何も押されていなければ，押されるまでチェックし続けます．何かキーが押されると，COMMAND＝？に戻ります．

このプログラムで，いろいろな絵を描いてみてください．さきほどは説明しませんでしたが，グラフィックマクロ命令の，UERFDGLHの前にBをつけると，ペンは移動しますが線は引きません．ちょうど，ペンを紙から離した状態になります．これも試してください．

```
run
COMMAND=? E50F50G50H50■
```

INPUT＄

さて，このプログラムで，違う図形を描かせるためにプログラムをいちいち変更する必要はなくなりましたが，１回１回画面が消えてしまうのが残念です．そこで，新しい入力命令INPUT＄を使ってみることにしましょう．

INPUT＄命令はINPUT命令に似ていますが，①［RETURN］を押す必要がない，②入力される文字数が決められる，③指定された文字数内であれば，どのような文字，記号でも入力できる，などの特徴があります．基本的なひな形は下のとおりです．

文字変数＝INPUT＄（文字数）

○カッコの中の文字数で何文字読むかを指定

○指定した文字数だけキーが押されたら，［RETURN］を押さなくても文字変数に入る

INPUT＄は，グラフィックモードでも，文字や数字を表示するテキストモードでも使えるので，なかなか便利です．

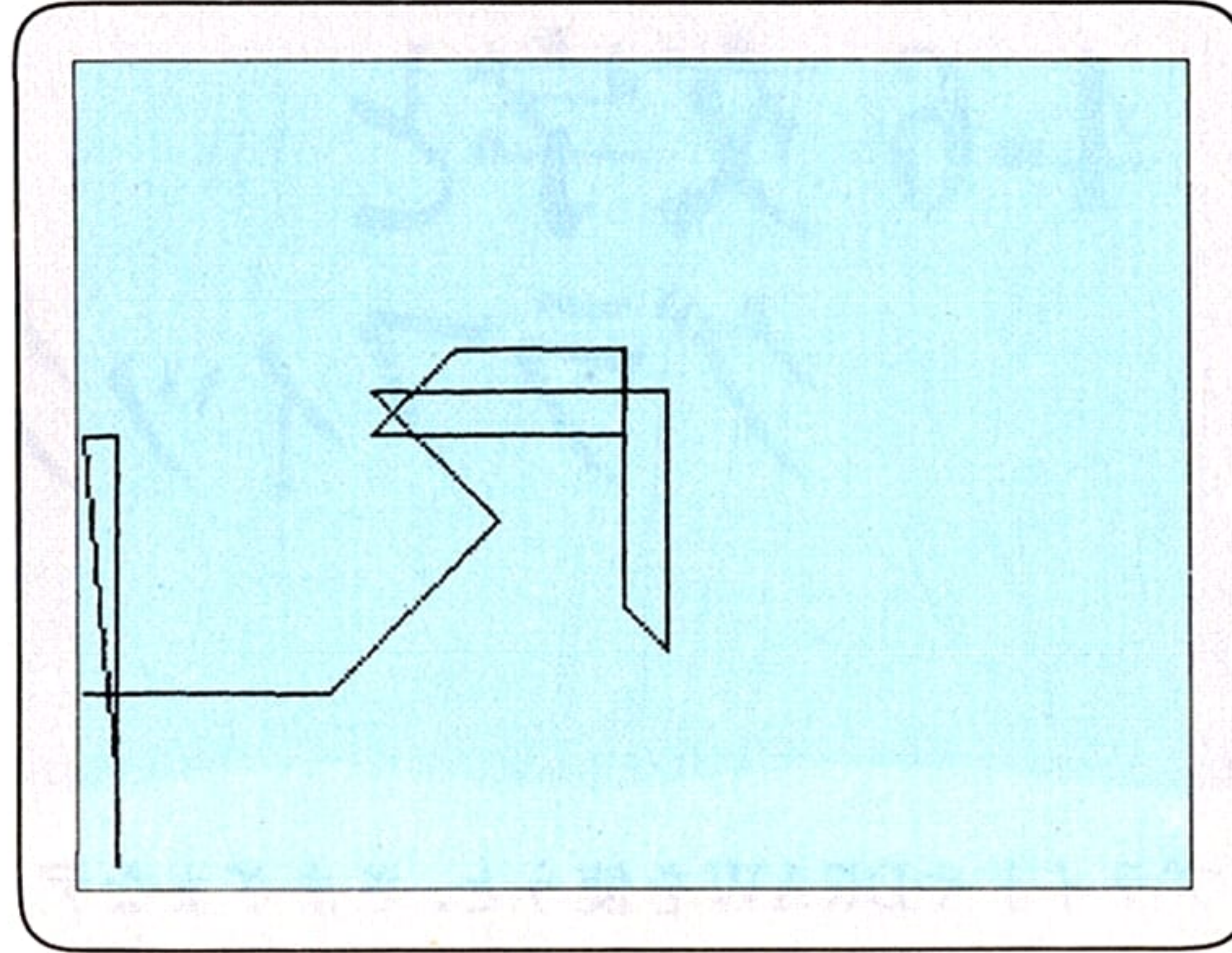
これを使ってプログラムを作ると次のようになります．

リスト

```
10 'DRAW
20 SCREEN 2
30 DRAW"BM128,96"
40 A$=INPUT$(1)
50 IF A$=CHR$(13) THEN 80
60 C$=C$+A$
70 GOTO 40
80 DRAW"XC$;"
90 C$=""
100 GOTO 40
```

このプログラムでは20行でグラフィック画面の設定をして，30行で最初のペンの位置を，画面のちょうど真中に持っていきます．ここまでは，いわば準備の部分です．次に40行でキーボードから1文字読み込んで，A＄という変数に蓄えます．そして，もしA＄に入った文字が，RETURN キーでなければ，C＄にA＄の内容を加えます．そしてまた40行で1文字読み込みます．さらにそれをC＄に足し，RETURN が押されるまで，40行でキーボードから読み込んだ文字をC＄に加えていきます．ここは，A＄で1文字ずつ読み込んだ文字を，C＄でグラフィックマクロ命令に組み立てている部分です．

RETURN が入力されたら，C＄に入っているグラフィックマクロ命令で絵を描きます(80行)．この入力のチェックの部分が50行です．ここでCHR＄(13)というのがあります．CHR＄は，キャラクタコードを文字に戻す命令で，RETURN のキャラクタコードは13番ですから，CHR＄(13)が RETURN キーを表していることになります．

このプログラムを打ち込んだら，実行してみましょう．実行しても何も表示されませんが，別にMSXが壊れているわけではありません．INPUT＄は，普通のINPUTと違って，？マークも出ませんが，気にせずグラフィックマクロ命令を，U50F20... のように打ち込んでいき，RETURN を押します．グラフィックマクロ命令を入力しているときも打ち込んだ文字は表示されませんので，注意して慎重に入力してください．RETURN を押すと，打ち込んだ命令が実行されます．

カーソル移動キーで絵を描く

最後に，カーソル移動キーを利用して自由に絵を描くプログラムを紹介しておきます．カーソルキーの動きをチェックするには，ちょっと工夫してINKEY＄を使ってみました．

プログラムはCHR＄を使った応用例にもなっています．CHR＄(30)は[↑]，CHR＄(29)は[←]，CHR＄(28)は[→]，CHR＄(31)は[↓]を表しています．このプログラムだとエラーが出てくることもなく，カーソルを動かす方向に線が描けます．もちろんSTICKを使って同じ機能のプログラムを作るのも簡単ですね．

リスト

```
100 'DRAW
110 SCREEN 2
120 DRAW"BM128,80"
130 K$=INKEY$
140 IF K$="" THEN 130
150 IF K$=CHR$(30) THEN A$="U1"
160 IF K$=CHR$(29) THEN A$="L1"
170 IF K$=CHR$(28) THEN A$="R1"
180 IF K$=CHR$(31) THEN A$="D1"
190 DRAW"XA$;"
200 A$=""
210 GOTO 130
```

MSX BASIC

4-6 文字とグラフィックの結合

これまで見てきたスプライトやDRAWを使うと，さまざまなデザインを作ることができると思います．

このsectionでは，グラフィックスに関するプログラムを作るときに有益なテクニックを紹介して，あなたがこれからプログラムを作るときの助けになるようにしましょう．

これまで，グラフィックスの命令を見てきて，グラフィック画面でできること，できないことが，おおよそわかってきました．グラフィックスでは，INPUTやPRINTが使えず，普通の文字や数字を表示するテキスト画面では，DRAWなどのグラフィックス用の命令は使えません．そうなると，１つの画面に，絵と文字の両方を表示できないので不便です．ゲームを作るときに，得点の表示もできないのでは困ります．なんとかして，文字をグラフィック画面で表示させる必要があります．

グラフィック画面に文字を書く

そこで，MSXには普通のPRINT命令以外に，グラフィックスの画面に文字を書く命令が用意されているのです．CHAPTER 3 でカセットテープに暗号のデータを書き込んだの

グラフィック画面に文字を書き込む方法

OPEN "GRP:" AS#1	…………グラフィック画面に文字を書く
▼	
PRINT#1, "ABC…	…………グラフィック画面への文字の書込み
▼	
CLOSE	…………グラフィック画面への文字の書込みを終える

を覚えているでしょうか.あのときカセットテープに書き込みをしたのと同じように,グラフィック画面に書き込みをすればよいのです.ここでもOPEN, PRINT#1, CLOSEを使います.

書き込む場所がグラフィック画面になっただけ,という程度に考えても問題ありません.OPEN文のところが少し変わるだけで,使い方はカセットテープのときと同じです.ただ,グラフィック画面の場合には,文字を表示する位置を指定する必要があるのです.

テキストモードで文字の表示位置を指定するときは,LOCATE命令を使いました.しかしOPEN "GRP:" とした中では,残念ながらLOCATEは使えません.そこでグラフィック画面での文字の表示位置指定には,PSETという命令を使います.

PSETは,もともとは画面上に点を打つ命令です.

```
PSET(横 , 縦) , 色
```

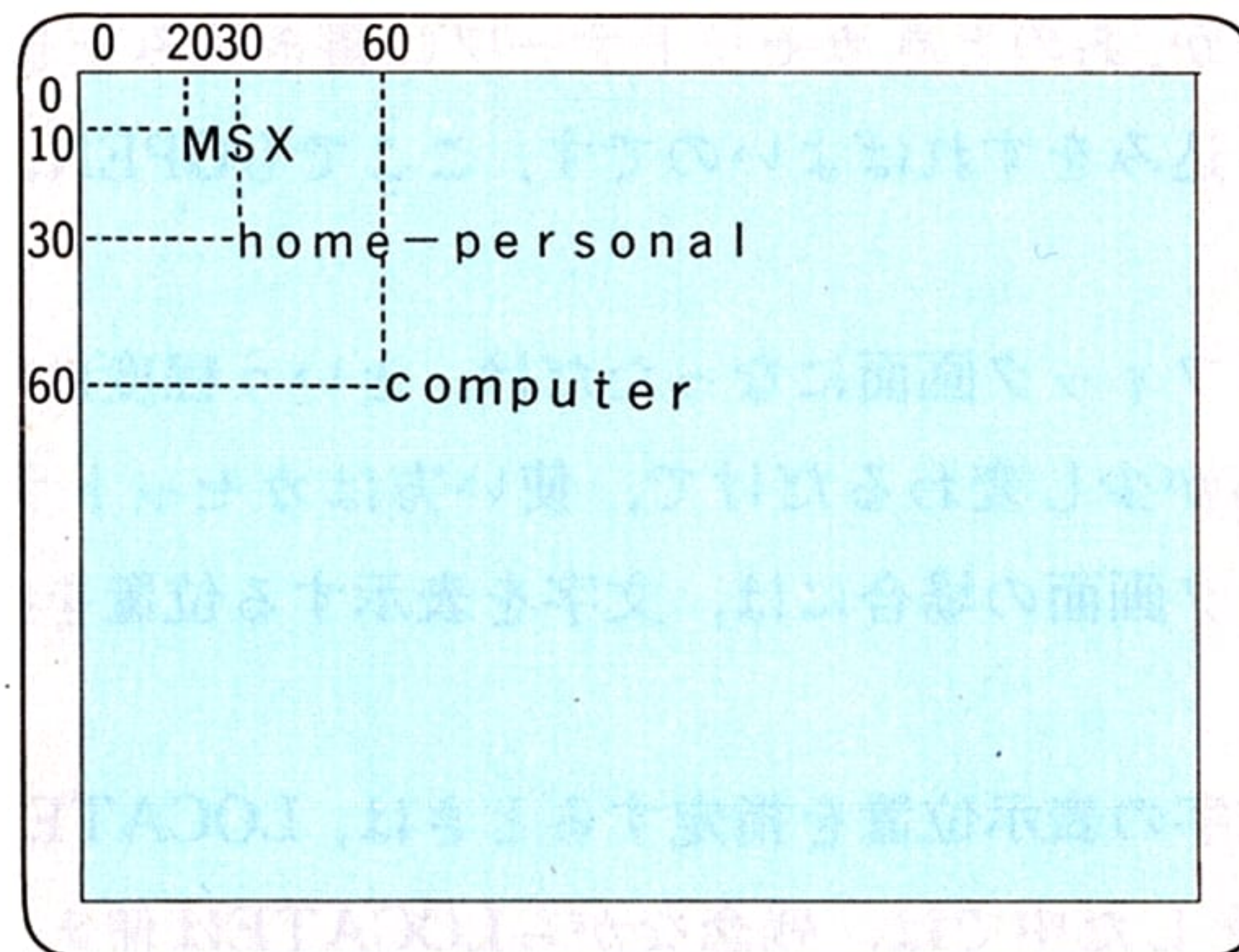
このPSET命令を実行すると,横と縦で指定した画面上の位置に,色で指定した点を1つ表示します.

つまり,本来は細かい点を打つことによって,絵や図を描くための命令なのですが,ここでは別の目的で使います.グラフィック画面での文字表示の位置は,最後に点が打たれたところから始まるようになっています.そのため,PSET命令を入れておけば,文字がその場所から表示されることになります.

しかしこの時,実際に点が打たれるのが見えるとみっともないので,ちょっと工夫します.画面の色をよく見ると,文字やグラフィックスの背景は,青色になっています.背景と同じ色の点を打てば,その点はまったく見えません.青のカラーコードは4ですから,文字を表示する位置を指定するときには,

```
PSET (X , Y) , 4
```

の型で使います.

サンプルとして,グラフィック画面に次のように文字を書いてみましょう.

これまでの説明でもわかるとおり，上のようにグラフィック画面に文字を書くプログラムは次のようになります．表示する位置，表示する文字などを変化させてみると，よく理解できるようになるでしょう．

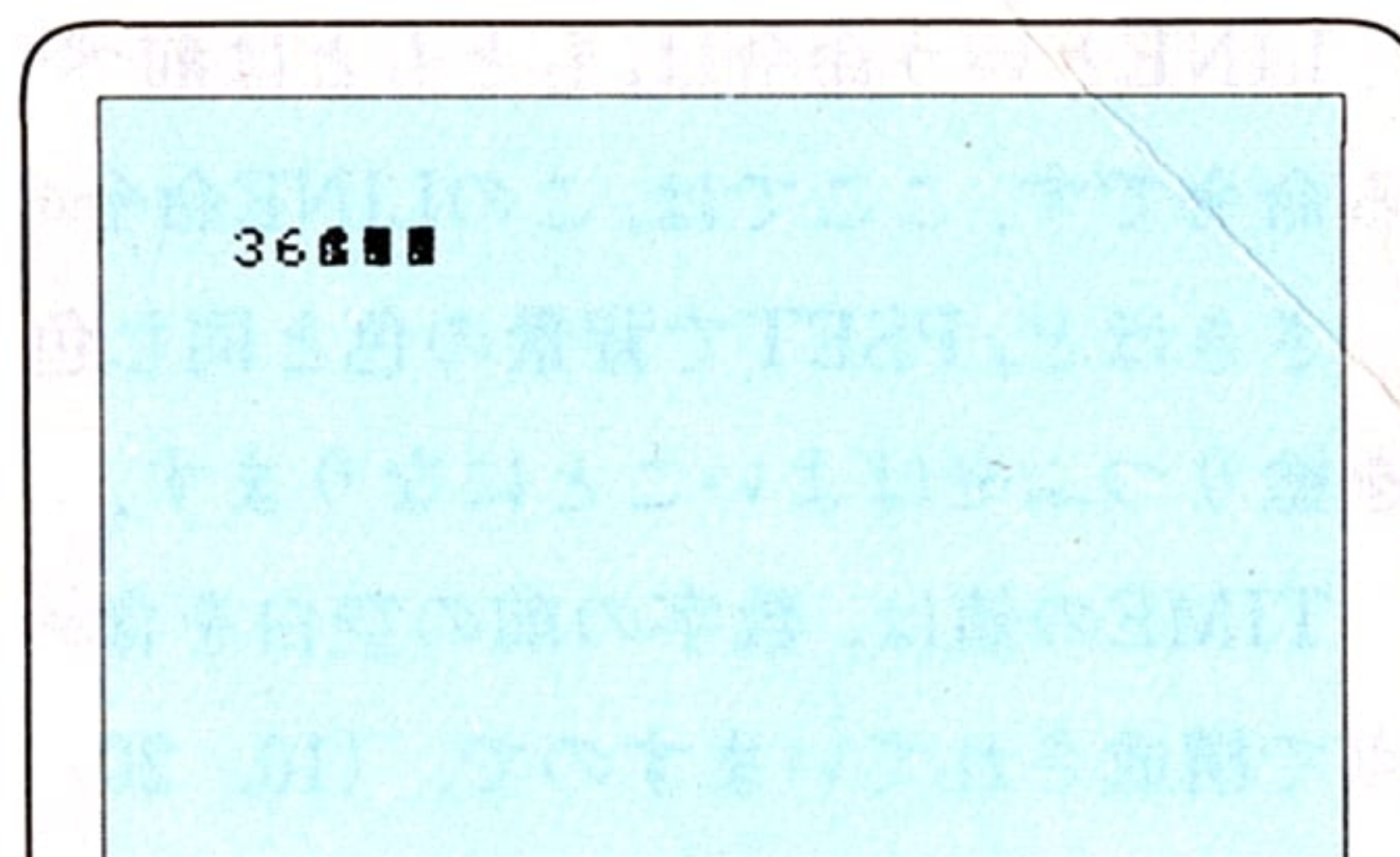
リスト

```
100 SCREEN 2
110 OPEN"GRP:" AS#1
120 PSET(20,10),4
130 PRINT#1,"MSX"
140 PSET(30,30),4
150 PRINT#1,"home-personal"
160 PSET(60,60),4
170 PRINT#1,"computer"
180 CLOSE
190 GOTO 190
```

タイマーを表示する

これで，グラフィック画面に文字を書き込む方法はわかりましたね．ところで，ゲームなどの得点を表示するときには，同じ場所に，何回も文字や数字を表示させたいことがあります．そのようなケースを想定して，グラフィック画面の横10，縦20の位置に，MSXのタイマーを表示させてみましょう．これまでの知識を使って考えると，プログラムは次のようになります．

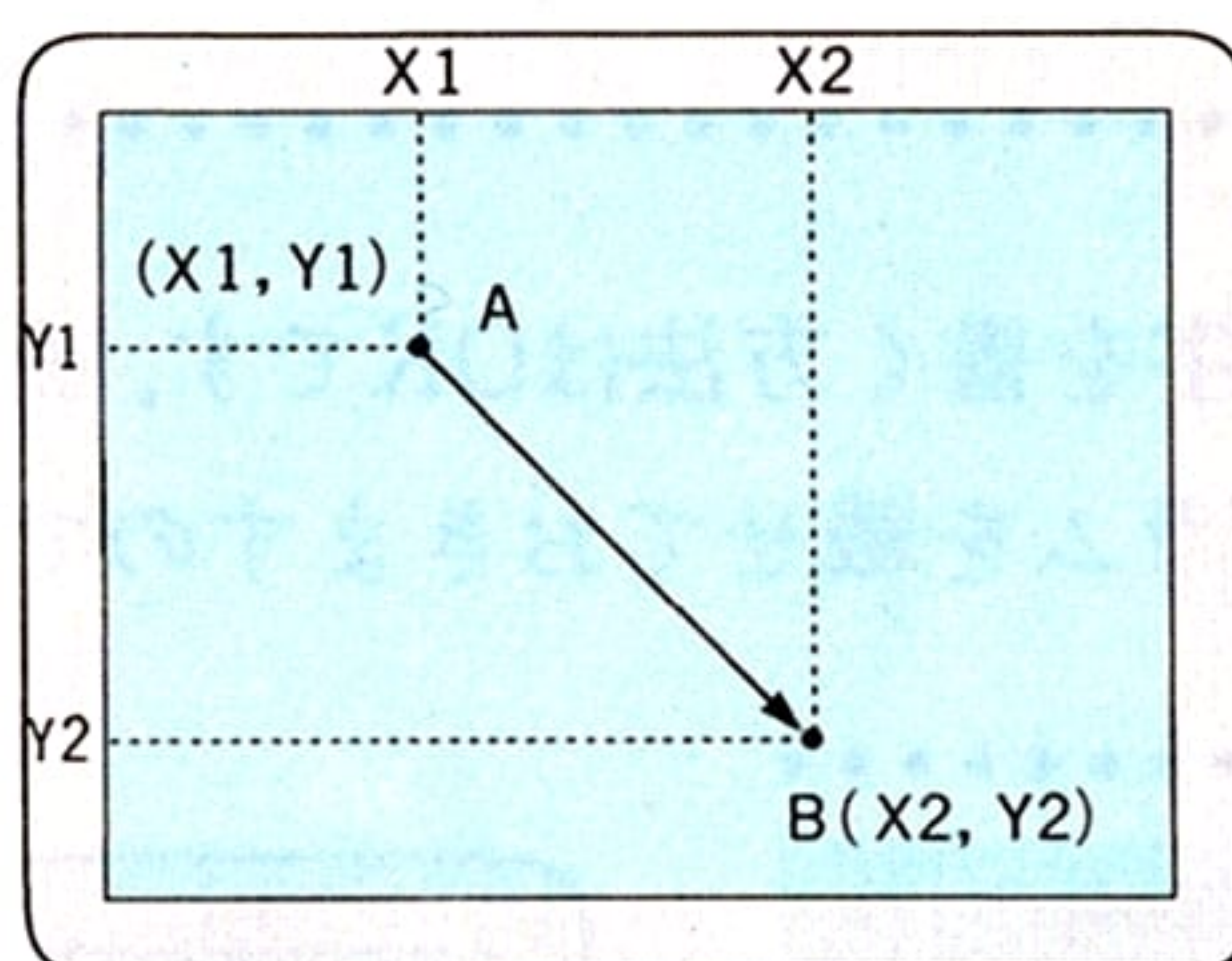
リスト

```
100 SCREEN 2
110 OPEN"GRP:" AS#1
120 PSET(10,20),4
130 PRINT#1,TIME
140 CLOSE:GOTO 110
```


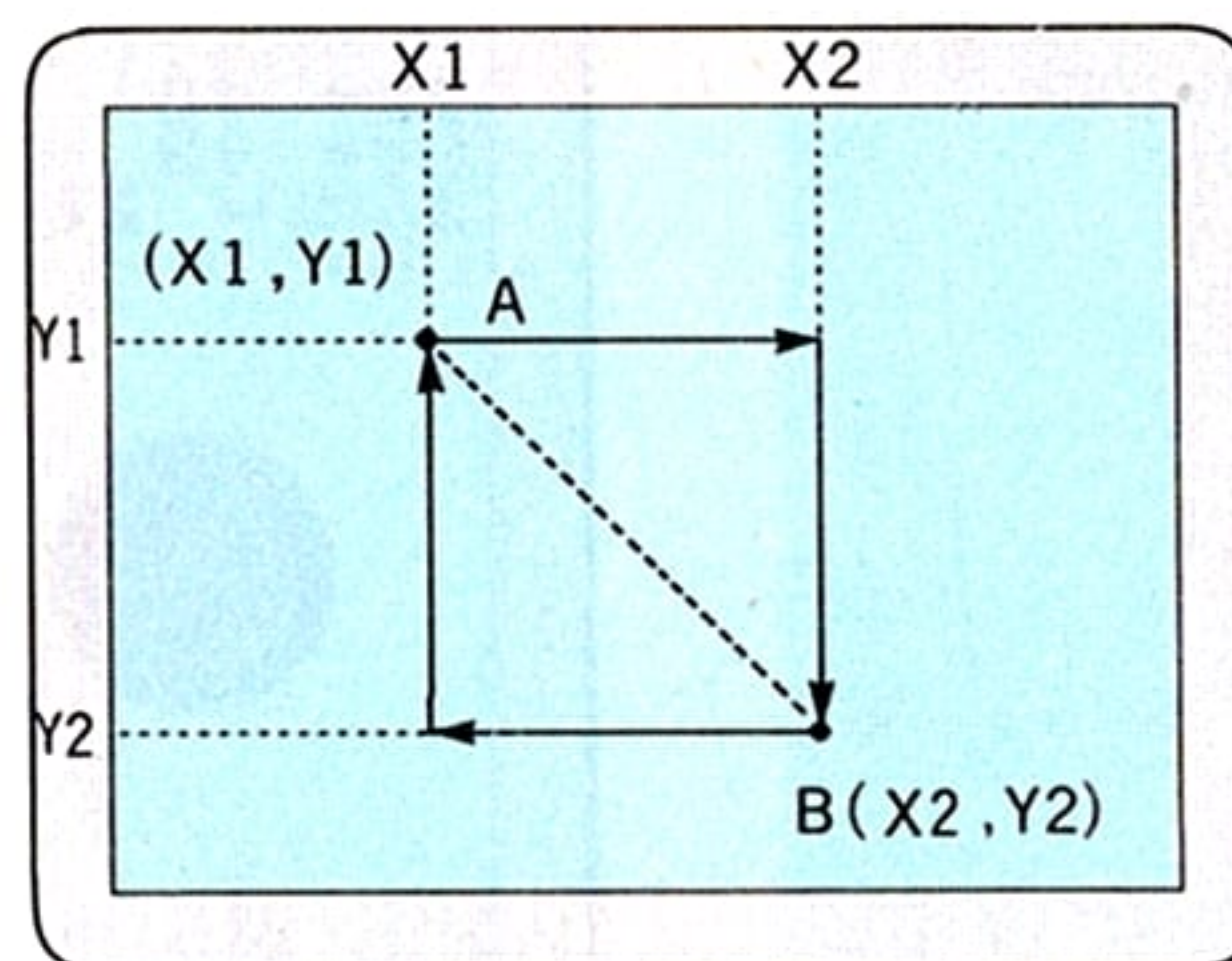
しかし，実際には数字が何重にも重なって読めなくなってしまいます．これを防ぐためには，文字を書く前にそれまであった文字を消して，そのあとで改めて書き直す作業が必要です．このようなときには，LINEという命令を使います．

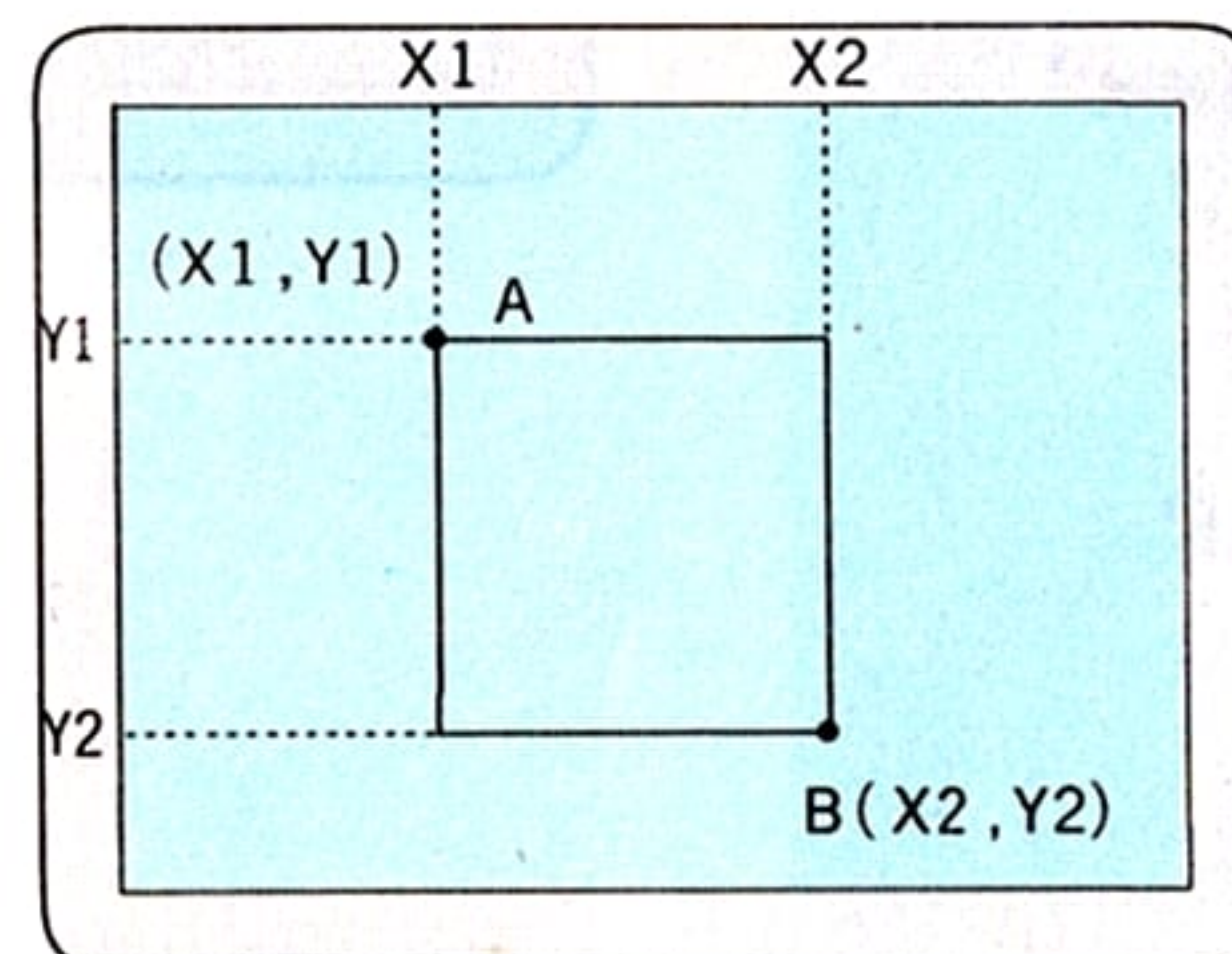
LINEの使い方

LINE　(X1,Y1)－(X2,Y2),C,　［なし／B／BF］

なし

点A(X1,Y1)と点B(X2,Y2)をむすぶC色の線を引く

X1，Y1は省略することができます．その場合は，直前のグラフィック画面に対して行った命令の座標(PSET，PRESETの座標，円の中心座標，直線の終端座標)がX1，Y1のかわりとなります

B

点Aと点Bをむすぶ直線を対角線とするC色の箱を描く

BF

Bのときと同じ箱を描いて，中をC色でぬりつぶす

LINEという命令は，もともとは前ページの説明のように，線を描いたり箱を描いたりする命令です．ここでは，このLINE命令の，箱を塗りつぶす機能を使って，文字を消します．

さきほど，PSETで背景の色と同じ色で点を打ったのと同じように，背景の色で四角形を塗りつぶせばよいことになります．

TIMEの値は，数字の前の空白を含めて，6ケタ必要としています．1文字は8×8の点で構成されていますので，(10，20)と(58，28)を対角線の頂点とする四角形を塗りつぶします．この部分を付け加えると，プログラムは下のようになります．

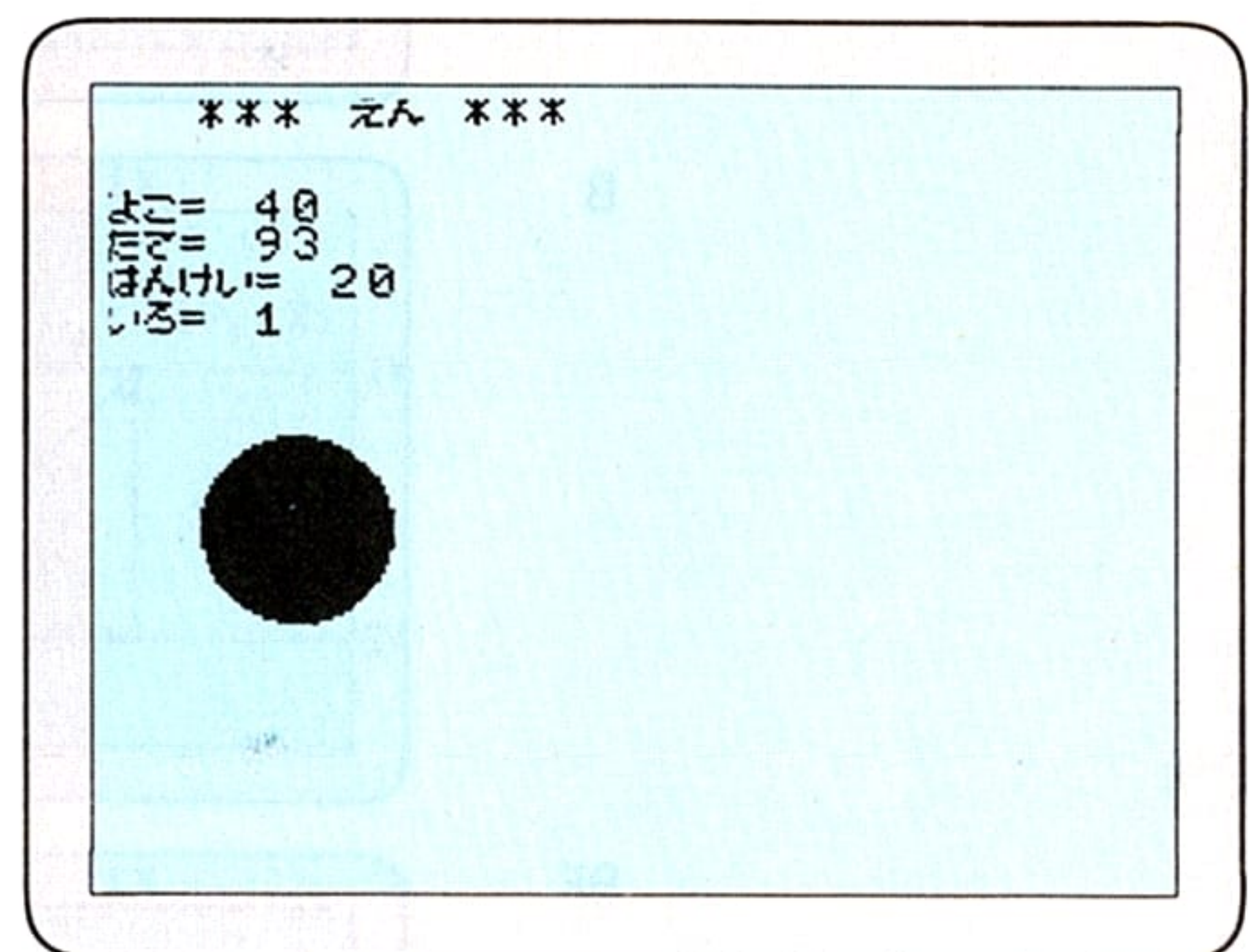
リスト

```
100 SCREEN 2
110 OPEN"GRP:" AS#1
115 LINE(10,20)-(58,28),4,BF
120 PSET(10,20),4
130 PRINT#1,TIME
140 CLOSE:GOTO 110
```

以上で，グラフィック画面に文字を書く方法はOKです．例として，グラフィックスと文字とが，両方表示されたプログラムを載せておきますので，参考にしてください．

リスト

```
10 'まる
20 INPUT"よこ=";X
30 INPUT"たて=";Y
40 INPUT"はんけい=";R
50 INPUT"いろ=";C
60 SCREEN 2
70 CIRCLE(X,Y),R,C
80 PAINT (X,Y),C
90 '
100 OPEN"GRP:" AS#1
110 PSET(20,0),4
120 PRINT#1,"*** えん ***"
130 PSET(0,20),4
140 PRINT#1," よこ=";X
150 PRINT#1," たて=";Y
160 PRINT#1," はんけい=";R
170 PRINT#1," いろ=";C
180 CLOSE
190 GOTO 190
```

```
*** えん ***

よこ= 40
たて= 93
はんけい= 20
いろ= 1
```

COLUMN

コンピュータ・グラフィックスの話

コンピュータで描いた絵や図のことをコンピュータ・グラフィックスと言います．ようするにコンピュータで視覚的な表現をすればすべてコンピュータ・グラフィックスになるわけです．

一見簡単そうなことですが，実際に思ったとおりに絵を表示させるのはかなり手間のかかることです．スプライト・パターンを使って傘ひとつ表示させるのにも，複雑な手続きを踏まねばならなかったことを考えるとよくわかると思います．

コンピュータは，絵や音を扱うのが苦手だと一般に言われますが，それはコンピュータが，主に数値や文字を処理する機械として発明されたからなのです．そして，ＭＳＸでも，大型コンピュータでも，コンピュータで絵や音を扱う場合は，実際には絵や音を数値に直して扱っているのです．

最近では，コンピュータ・グラフィックスを使ったテレビCMをひんぱんに見かけるようになりました．テレビCMのそれは，質感や重量感が出ていたり，空間構成が美しかったり，思いがけない動きをしたりと，とてもコンピュータで作ったとは思えないほどすばらしいものですが，そのおおもとは，やはり数値の処理によって作られているのです．

最前線のコンピュータ・グラフィックスがＭＳＸと違っている点は，色が何千色も使えたり，絵を構成する一つひとつの点が目に見えないほど細かかったり，ということです．そして，それらの要素を含んだ気の遠くなるような大量の絵の数値データを，ＭＳＸの何千倍も処理の速いコンピュータを使って，何日間も計算して作っているのです．

MSX BASIC

4-7 コンピュータミュージック入門

CHAPTER 4のこれまでのsectionでは，使う人の視覚に訴えるプログラムを，スプライトやDRAWといった機能を使って作ってきました．しかし，MSXの機能はまだまだたくさんあります．これまでに紹介したものは，その中のほんの一部にすぎません．

さてこれからしばらくは，聴覚に訴えるプログラムを作ってみることにしましょう．コンピュータで音を使う用途として，曲を演奏することと，ゲームなどで効果音を出すことが挙げられます．これから，この2つの課題を追求していきましょう．

音を出す

このsectionでは，まず普通に曲を演奏させてみましょう．MSX　BASICには，曲を演奏させるための専用の命令が用意されているので，音の高さと長さを指定するだけで，自動的に曲の演奏をしてくれます．音を出す命令は，PLAYです．PLAYの型は，次に示すとおりです．

PLAY〝ミュージックマクロ命令〞

ミュージックマクロ命令というのは，演奏する曲の音の高さ，長さ，演奏の速さなどを指示するものです．DRAWの中で具体的な絵の描き方を指示したグラフィックマクロ命令と同じようなものです．それでは，ミュージックマクロ命令の詳細を説明しましょう．

音の高さ

ミュージックマクロ命令で指定する一番基本的な要素は，ド・レ・ミ・ファ……のどの音を鳴らすのか，ということです．つまり，音階の指定が一番基本です．これはド・レ・ミ・ファ……とせずに，次のように，C・D・E・F……とアルファベットで指示します．

楽譜とミュージックマクロ命令との対応

普通の呼び方……	ド	レ	ミ	ファ	ソ	ラ	シ	休符
ミュージック……マクロ命令	C	D	E	F	G	A	B	R

半音の表し方

ソ	ソ♯	ソ♭
G	G＋	G−

♯（半音高い）……音符を表すアルファベットの後に＋を付ける
♭（半音低い）……音符を表すアルファベットの後に−を付ける

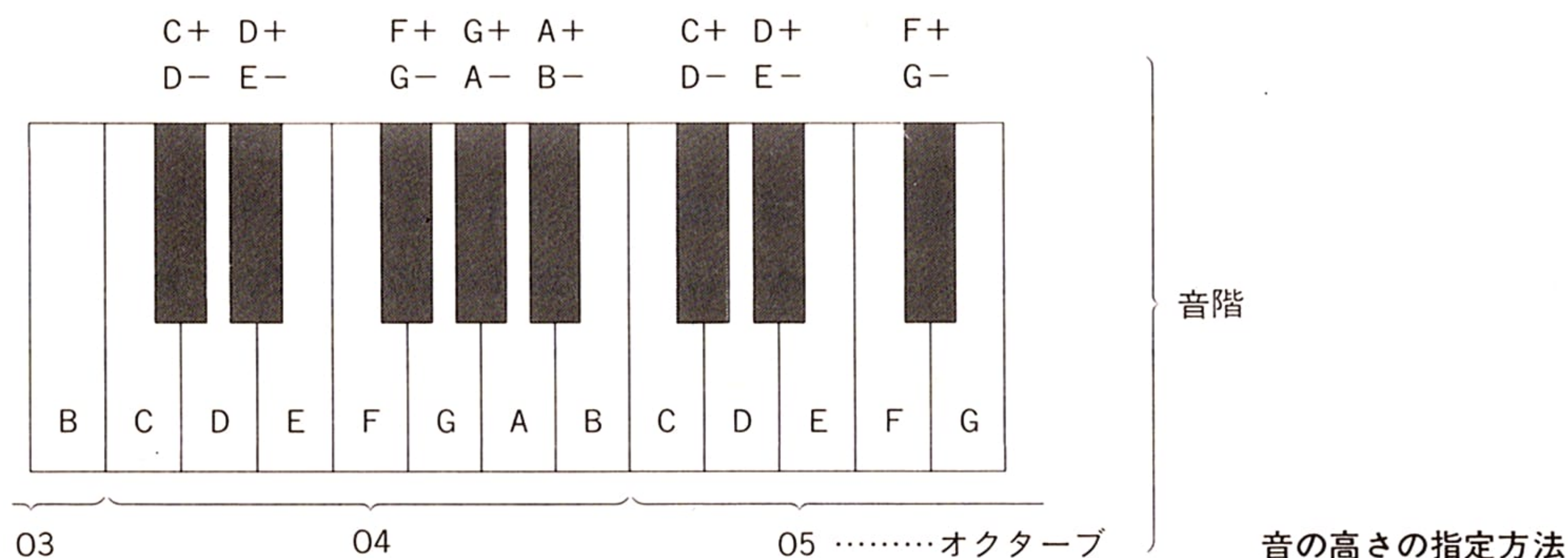
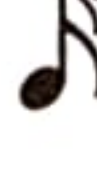
ドの音を鳴らすときは，PLAY "C"，ドミソと鳴らしたいときは，PLAY "CEG" とすればよいのです．

ところで，このままでは音域が狭すぎて話になりません．MSX は 8 オクターブもの広い範囲で音が出せるので，音域を広げてみましょう．オクターブの指定には，アルファベットのOを使います．O1～O8の範囲で，オクターブの指定をします．この数字は，小さいほど音が低く，大きいほど高くなっていますので，O1が一番低く，O8が一番高いことになります．なお，電源を入れた状態ではO4にセットされています．オクターブの指定は，一度指定すると，次の指定があるまではそのままです．

例）PLAY "O4CDEFGABO5C"

このようにすると，ドレミファソラシドと，ちゃんと音階を鳴らします．音の高さに関する指定はこれだけです．

C+ D+ F+ G+ A+ C+ D+ F+
D− E− G− A− B− D− E− G−
B C D E F G A B C D E F G
O3 O4 O5 ………オクターブ
音階

音の高さの指定方法

音の長さ

先の例では音の長さは一定です．曲を演奏するためには音の長さも自由に指定できなければなりません．長さの指定はLで行います．

電源を入れたばかりの状態では，音の長さはL4に指定されています．

音の長さの指定方法（Lnを使って）

n：（1～64の間の整数）音の長さをn分音符にする

例）　L4 ……… ♩（4分音符）
　　　L8 ……… ♪（8分音符）
　　　L16……… 𝅘𝅥𝅯（16分音符）
　　　L32……… 𝅘𝅥𝅰（32分音符）

長さをLで指定すると，次に指定があるまで，原則として以後の音はその長さで演奏されます．しかし，曲全部の音符が同じ長さということはめったにありません．音の長さを1音だけ変えるためには，音程のあとに，そのまま長さを指定する数字を加えます．たとえば，L8と指定した曲で，4分音符がひとつ出てきたときは，PLAY〝‥C4D‥〟のようにすれば，このCの音だけ4分音符になります．

また，休符はRで表しますが，休符で休む長さはLで指定した長さに左右されないので，R8，R96などのように，いちいち長さを指定しなければなりません．ただし，数字を省略したときにはR4とみなされます．また，楽譜の中でよく付点が付いているものがあります．このようなときは，こちらも付点を付け，F4．のように書きます．

曲の速さ

同じ曲でも，全体として速く演奏したり遅く演奏したりすることがあります．曲全体の速さは，Tnの型で指定できます．この場合nは，32～256の間の数字で，1分間に演奏される4分音符の数を指定します．よく楽譜をみると，一番左上に「＝55」などと書かれていますが，nにはこの数字を書けばよいのです．

楽譜によっては，Moderatoとか，Andanteと，文字で速さを指定しているものもあります．これらとnの対応は次のとおりです．

速度標語	速さ	Tnの値
Largo（ラルゴ）	♩≒ 46	46
Adagio（アダージョ）	♩≒ 58	58
Andante（アンダンテ）	♩≒ 72	72
Andantino（アンダンティーノ）	♩≒ 80	80
Moderato（モデラート）	♩≒ 92	92
Allegretto（アレグレット）	♩≒108	108
Allegro（アレグロ）	♩≒132	132

速度標語とｎの関係

エリーゼのために

それではいよいよ，「エリーゼのために」の冒頭部分を演奏させてみましょう．楽譜は下のとおりです．

Für Elise

エリーゼのために

L.v.Beethoven

Poco moto

これを，今覚えたばかりの規則でプログラムに直してみると，次のようになります．

音を間違えた場合，他のものと違って，目でよく確かめるわけにもいかないので，修正がしやすいように，データにドレミファを振っておきましたので参考にしてください．

```
100 'fuer elise ………タイトル
110 PLAY"T80" ………速さを♩=80に
120 PLAY"L16" ………音の長さを♪に
130 PLAY"O5ED+"
          ミレ#
    オクターブ5に指定
140 PLAY"ED+EO4BO5DC"
         ミレ#ミ シ レド
    オクターブ4 オクターブ5
150 PLAY"O4A8R16CEA"
         ラ  𝄿  ドミラ
    オクターブ4
160 PLAY"B8R16EG+B"
         シ 𝄿  ミソ#シ
170 PLAY"O5C8R16O4EO5ED+"
         ド  𝄿   ミ  ミレ#
    オクターブ5 オクターブ4 オクターブ5
180 PLAY"ED+EO4BO5DC"
         ミレ#ミ シ レド
    オクターブ4 オクターブ5
190 PLAY"O4A8R16CEA"
         ラ  𝄿  ドミラ
    オクターブ4
200 PLAY"B8R16O4EO5CO4B"
         シ 𝄿   ミ ド シ
    オクターブ4 オクターブ5 オクターブ4
210 PLAY"A4"
         ラ
```

さて，このプログラムを見ると，PLAYが並んでいます．曲を変えるときには，プログラムを打ち直す必要があります．修正ができるだけ少なくて済むようにプログラムを工夫してみましょう．

リスト

```
100 'fuer elise
110 READ A$
120 IF A$="¥" THEN END
130 PLAY A$
140 GOTO 110
150 DATA T80
160 DATA L16
170 DATA O5ED+
180 DATA ED+EO4BO5DC
190 DATA O4A8R16CEA
200 DATA B8R16EG+B
210 DATA O5C8R16O4EO5ED+
220 DATA ED+EO4BO5DC
230 DATA O4A8R16CEA
240 DATA B8R16O4EO5CO4B
250 DATA A4
260 DATA ¥
```

このプログラムでは，スプライトのところで説明した，READ～DATAを使って，音楽のデータをプログラムの中に蓄えています．音楽のデータを，文字変数A＄に読み込んで，PLAY　A＄の型で演奏しているのです．これでわかるとおり，PLAYは直接音楽のデータを指定するだけでなく，変数の形でも指定できます．データを読み，演奏する．これを繰り返すのですが，そのままで演奏させていくと，音楽のデータがつきてしまい，Out of DATAというエラーが出てしまいます．そこで，データの終わりを示す印を付けておく必要があります．プログラムの中で，終わりの印をチェックしてこの印が出てきたらデータを読むのをや

めるようにします．このプログラムでは終わりの印として，¥を付けています．

これで，汎用の音楽演奏プログラムが完成です．DATAの部分を変えて，いろいろな曲を演奏させてみてください．

和音の演奏

これまでは，一度に1つの音だけを出してきましたが，MSXでは同時に3つまでの音を出すことができるので和音にしたり，オブリガードを付けることができます．そのためには，PLAYを次のような型で使います．

PLAYによる和音の出し方

PLAY "音のデータ"，"音のデータ"，"音のデータ"

パート① パート② パート③

上のように3つある旋律を，［，］で区切って並べると，同時に3つの音が出せます．試しに，PLAY "C"，"E"，"G" としてみてください．ドミソの和音が出たはずです．この，同時に違う音が出るという機能を使って，さきほどの「エリーゼのために」に伴奏を付けてみました．あなたもMSXで，いろいろな曲を演奏してみてください．

リスト

```
100 'fuer elise
110 READ A$,B$
120 IF A$="¥" THEN END
130 PLAY A$,B$
140 GOTO 110
150 DATA T80,T80
160 DATA L16,L16
170 DATA O5ED+,R8
180 DATA ED+EO4BO5DC,R4R8
190 DATA O4A8R16CEA,O2AO3EAR16R8
200 DATA B8R16EG+B,O2EO3EG+R16R8
210 DATA O5C8R16O4EO5ED+,O2AO3EAR16R8
220 DATA ED+EO4BO5DC,R4R8
230 DATA O4A8R16CEA,O2AO3EAR16R8
240 DATA B8R16O4EO5CO4B,O2EO3EG+R16R8
250 DATA A4,O4C4
260 DATA ¥,¥
```

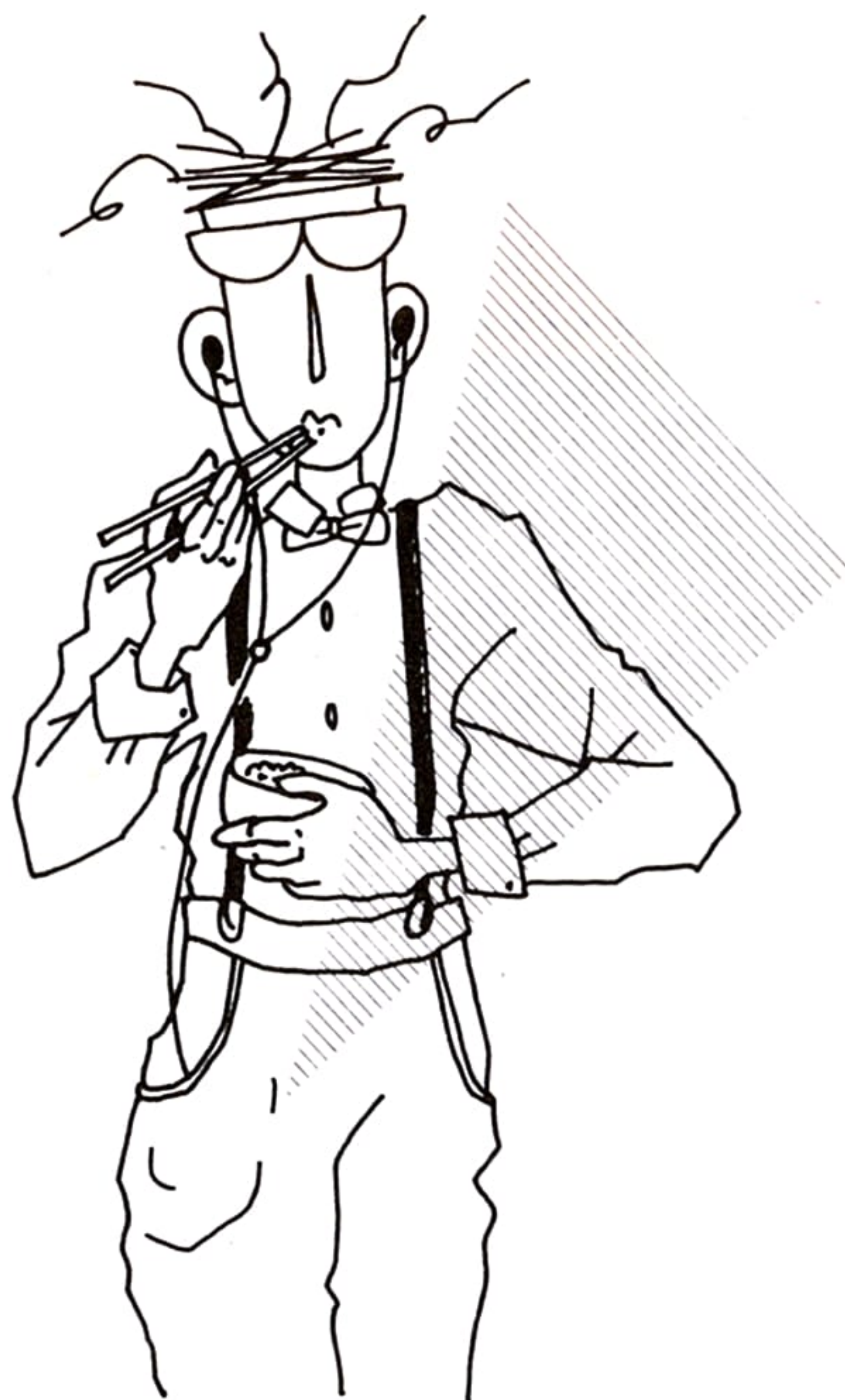
MSX BASIC

4-8 効果音を付けよう

音楽の演奏の方法はわかりましたか？　楽譜が読める人ならば，試行錯誤していくうちに理解できるようになったと思います．残念ながら，自分の音楽レベルはカラオケで歌ったり，好きなメロディーを口ずさむ程度だという人は，ちょっと苦労するかもしれません．どうしても，自分で音のデータが作れない人は，音楽のわかる友達にやってもらうのもよいでしょうし，これからMSX用の音楽用ソフトが出てきますので，そういったものを利用してもよいでしょう．

さて，次に効果音を使う方法を研究してみましょう．みなさんがゲームをするときに，音がなかったらどうでしょうか．おそらくまったく同じゲームでもおもしろさは半減してしまうでしょう．さきほど作ったスロットマシンゲームも，効果音が入ったらもっと楽しくなるかもしれません．

ゲームの効果音とはちょっと違いますが，シンセサイザーの真似ごとも，できればしてみたいものです．「エリーゼのために」も，今のままではオルガンで演奏しているみたいですね．いろいろと音色を変えて，演奏させてみたいものです．

ここでは，ゲームに使う効果音と，シンセサイザーのように音色を変える，という2つのことに挑戦してみましょう．

効果音を鳴らす

効果音といっても，音には違いないので，なにも難しく考える必要はありません．一番単純な方法は，ただひたすら速く鳴らすことです．下のリストを見て，すぐに打ち込んでください．

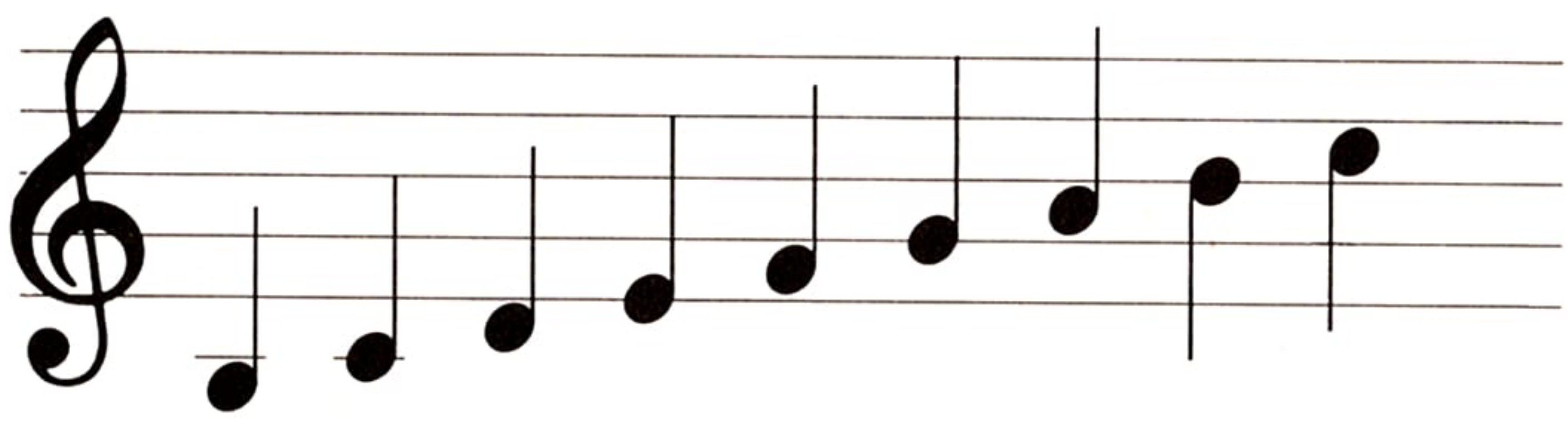
リスト

```
10 PLAY"T255L64"
20 FOR I=1 TO 100
30  PLAY"O7CD"
40 NEXT I
```

残念ながら本の上では何も聞こえてきませんが，ピロピロピロ……と音がするはずです．普通の音を使って効果音を出す方法としては，

① ひたすら速く音を繰り返す
② 短い音で，音を繰り返す
③ 極端に高い(もしくは低い)音を繰り返す

といったところが考えられます．上のリストの音を低くして試してみましょう．

リスト

```
10 PLAY"T255L64"
20 FOR I=1 TO 100
30  PLAY"O1CD"
40 NEXT I
```

もうひとつの音の指定

効果音の一番単純な例として，音をひたすら速く繰り返して，極端に高い音を出してみました．このようなときに便利なように，MSXでは，音の高さをすべて数字で指定できるようになっています．これまで，音の高さをオクターブで示すOと，音階で示すCDEFなどで表してきました．これに対してMSXでは，8オクターブのすべての音に，1～96の番号が付いていて，次のような型で音の高さを指定することもできます．

数字による音の高さの指定方法

これまでの表し方	B	C	D	E	F	G	A	B	C
オクターブ	O3	O4							O5
Nによる高さの指定	35	36	38	40	41	43	45	47	48

①直接指定する
　Nn ➡ nは音の高さを表す数字（0～96）
　例）N1………O1のD(レ)
　　　N95………O8のB♯(シのシャープ)
　　　N0………R(休符)

②変数で指定する
　N＝I；➡ Iは音の高さを表す変数
　　　　　Iは0～96の間で指定

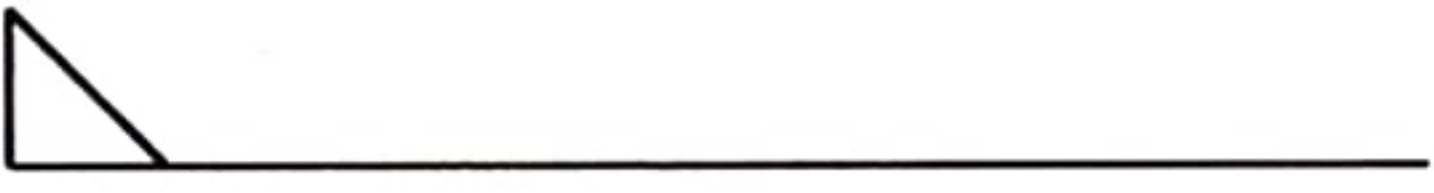
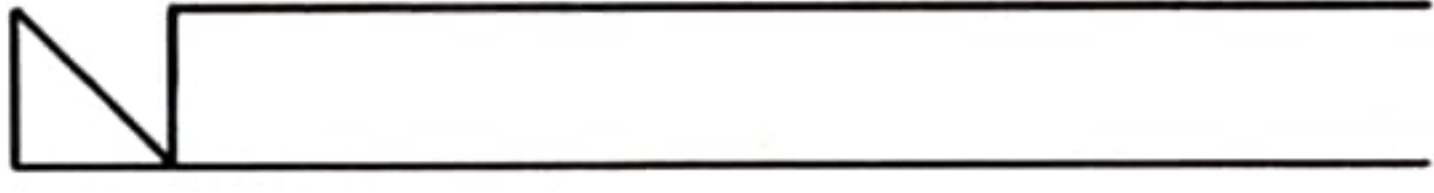
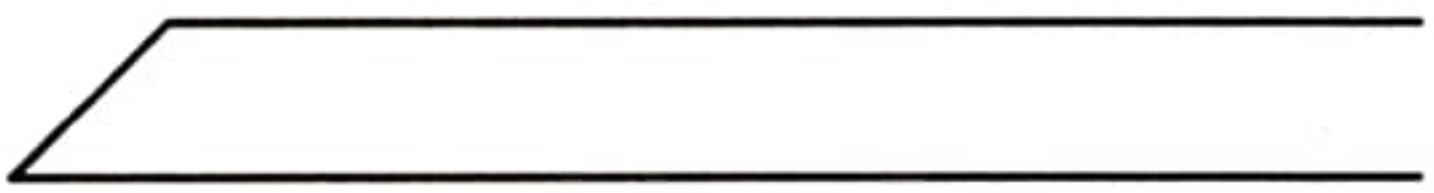
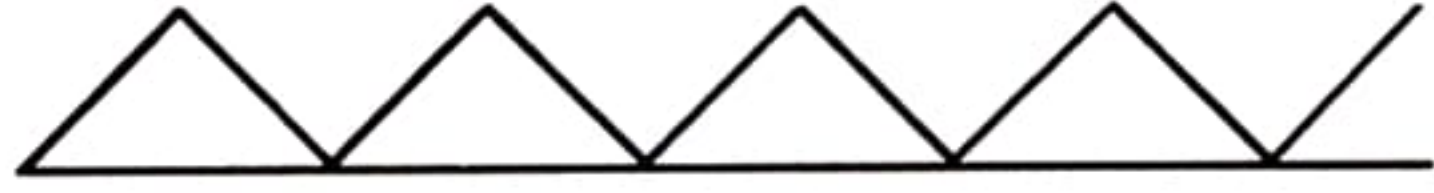
特に②の型がとても重宝します．たとえば次のプログラムを見てください．

リスト

```
10 PLAY"T255L64"
20 FOR I=1 TO 96
30  PLAY"N=I;"
40 NEXT I
```

このプログラムは，MSXで出せる一番低い音から，一番高い音までを，連続して鳴らすものです．もし，これを普通の形で書いたら，PLAY " O1DEFGABO2CDEFGABO3....”と大変なことになります．

音を数字で表せるということは，乱数を使って，音を指定することができるわけです．この機能は，効果音を作るときに便利です．次のプログラムで，その便利さを確かめてください．

リスト

```
10 PLAY"T255L64"
20 FOR I=1 TO 100
30  A=INT(RND(1)*20+76)
40  PLAY"N=A;"
50 NEXT I
```

エンベロープ

これまでに見てきたような単純な方法の他に，MSXでは，音色そのものを変えることができます．今まで出していた音は，ごく単調な一定した音でしたが，その音にすばやく音量の変化を加えると，音色は変わってきます．それがエンベロープです．音量を変化させる形を指定するのがS，周期を指定するのがMという文字です．

MSXが扱えるSの型は次のとおりです．8種ある中から，図の左の番号で形を選びます．

Sの値	音の波の形	Sの値	音の波の形
0～3 9		11	
4～7 15		12	
8		13	
10		14	

Sの値と音の波の形の対応

M(周期)

周期の指定はMを使い，M(数字)の型で指定します．周期といってもわかりづらいかもしれません．変化の速さを表すもの，といった程度に覚えておけばよいでしょう．(数字)は0～65535の間で指定します．

SとMの組み合わせで，たくさんの音色の音を出すことができます．次に挙げるものは，そのほんの一例です．

リスト

```
10 PLAY"S9M3000CEG"
20 PLAY"S9M1000CEG"
30 PLAY"S9M5000CEG"
40 PLAY"S7M1000CEG"
50 PLAY"S4M500CEG"
60 PLAY"S8M500CEG"
```

これを使って,「エリーゼのために」の音色をもう少しピアノに近づけてみましょう.波形をS0やS9にすると,ピアノに似た感じの音になります.

ここでは,S9M8000としてみました.

リスト

```
100 'fuer elise
110 READ A$,B$
120 IF A$="¥" THEN END
130 PLAY A$,B$
140 GOTO 110
145 DATA S9M8000,S9M8000
150 DATA T80,T80
160 DATA L16,L16
170 DATA O5ED+,R8
180 DATA ED+EO4BO5DC,R4R8
190 DATA O4A8R16CEA,O2AO3EAR16R8
200 DATA B8R16EG+B,O2EO3EG+R16R8
210 DATA O5C8R16O4EO5ED+,O2AO3EAR16R8
220 DATA ED+EO4BO5DC,R4R8
230 DATA O4A8R16CEA,O2AO3EAR16R8
240 DATA B8R16O4EO5CO4B,O2EO3EG+R16R8
250 DATA A4,O4C4
260 DATA ¥,¥
```

どうでしょう.さきほどよりずっとピアノに近い音になりましたね.Mの値を小さくすれば,もっとスタッカートがきいて,こまぎれの音になりますし,逆に大きくすると音がいつまでも残ります.

Sの値を変えてみるとさらにおもしろい音色になります.たとえば,S8M500くらいで演奏するとマンドリンのような音になります.

いよいよSOUNDへ

これまで，音を出す命令としてはPLAYの一本槍できました．しかし，音を出すための命令にはこのPLAYの他にもSOUNDという命令があります．SOUNDは，これこそ効果音づくりのための命令で，ピストルを撃つ音，飛行機の爆音，ミサイルの発射音など，多くの種類の音が，リアルに作り出せるのです．

次のsectionではSOUNDの例とスプライトの復習を兼ねて，ロケットを飛ばすプログラムを作ってみます．

COLUMN

音楽とコンピュータ

音楽とコンピュータとの関わりは，コンピュータが実用化されてすぐに始まっています．まず最初に目をつけたのは，現代音楽の作曲家たちでした．1950年代後半，彼らは確率統計の考え方を作曲の世界にもち込み，その計算のためにコンピュータを使ったのです．

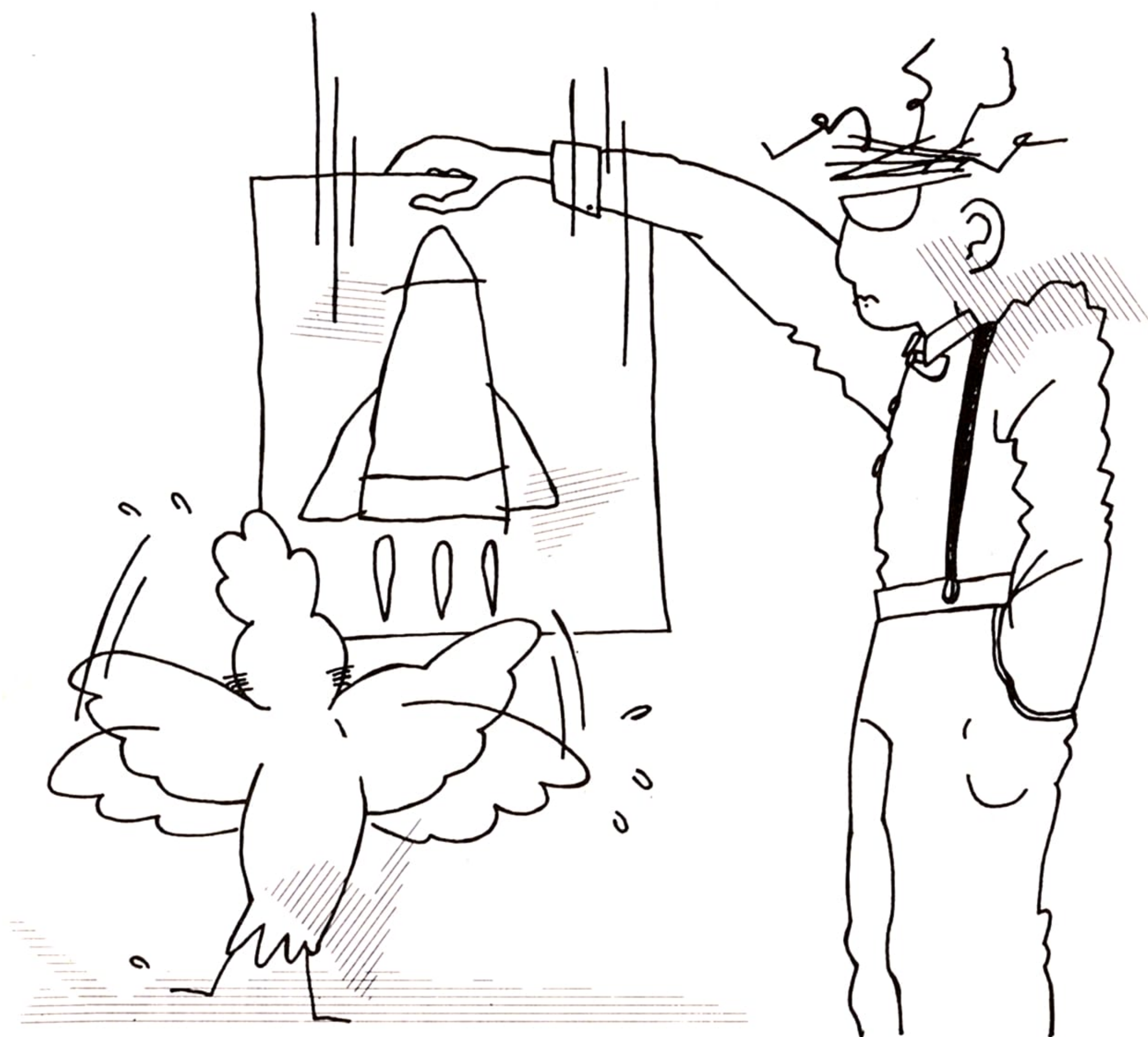
1960年代に入ると，コンピュータの機能もあがり，エレクトーンやシンセサイザーなど電子楽器の原型もできあがりました．コンピュータは電子楽器に直接つながれ，演奏を行うようになったのですが，この時期に，音楽にコンピュータを使うことは大変実験的なことでした．

1970年代に入るとコンピュータは少し前衛的な人々のあいだで徐々に音楽に使われるようになりました．そして70年代後半，コンピュータは音楽にいっせいに使われ出したのです．その背景には，安価なマイクロコンピュータの登場や，電子楽器の高度な発展がありました．

テクノポップなどのように，正確無比なリズムや無機質な音を好んで使い，コンピュータのもつインダストリアルなイメージを音楽性の一部として取り入れていったのも，つい最近のことです．

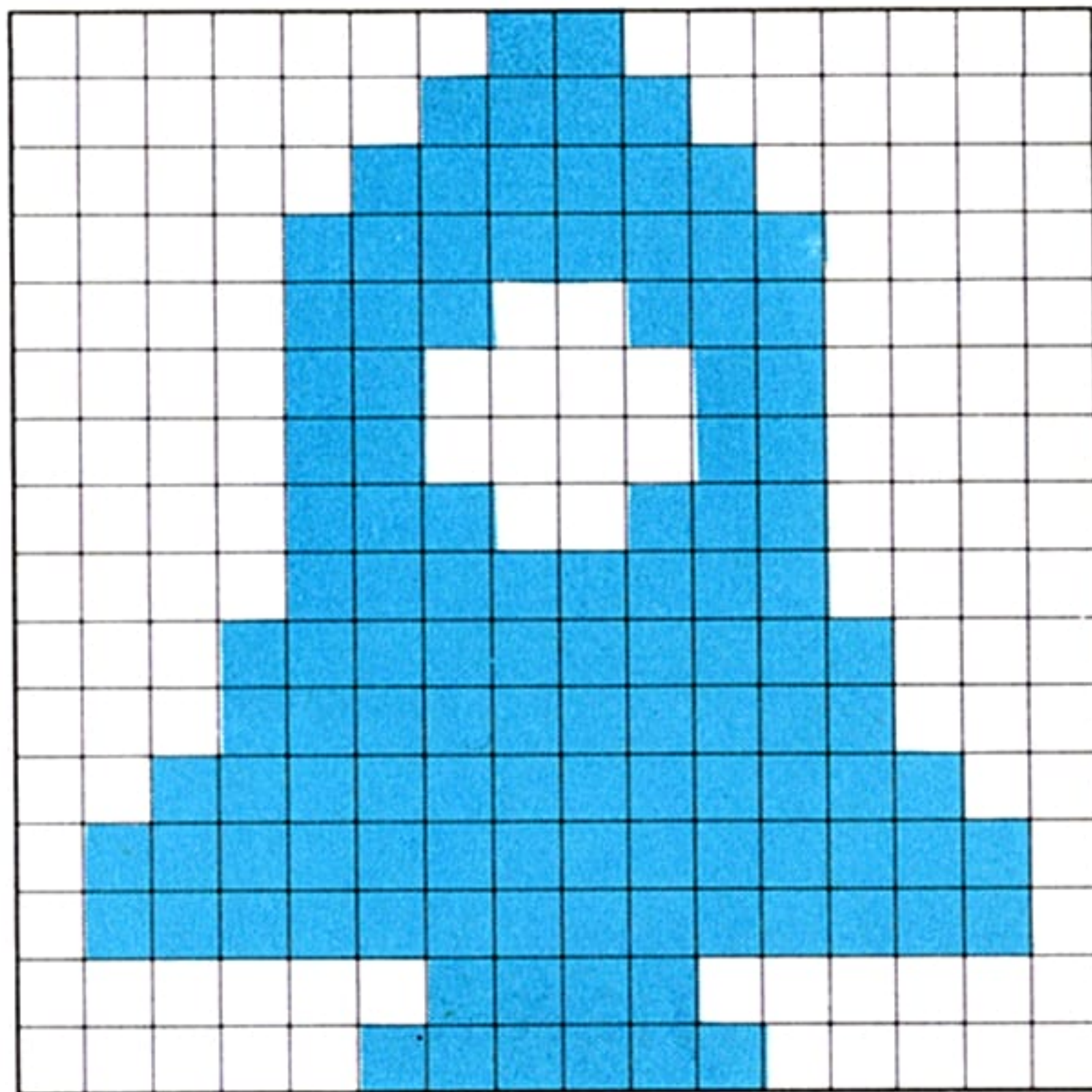
MSX BASIC

4-9 ロケットが飛ぶ

ロケットの型をデザインして，スプライトパターンとして登録し，それをカーソルで動かすようにしましょう．ただ動くだけではおもしろくないので，移動する方向に応じて色が変わり，さまざまなロケットの噴射音が聞こえるようにしてみました．まずは，全体のリストを見てください．

リスト

```
100 SCREEN 1,3:COLOR 1,2
110 C$=""
120 C=15
130 FOR I=1 TO 32
140  READ A
150  C$=C$+CHR$(A)
160 NEXT I
170 SPRITE$(0)=C$
180 FOR I=0 TO 13
190  SOUND I,0
200 NEXT I
210 '
220 X=120:Y=160
230 '
240 SOUND 7,255
250 PUT SPRITE0,(X,Y),C,0
260 ON STICK(0) GOTO 330,340,350,360,370,380,390,400
270 SOUND 7,255
280 S=2
290 X=X+2-INT(RND(1)*5)
300 Y=Y+2-INT(RND(1)*5)
310 C=15
320 GOTO 250
330 Y=Y-S          :C=1:GOTO 410
340 Y=Y-S:X=X+S:C=9:GOTO 410
350       X=X+S:C=3:GOTO 410
360 Y=Y+S:X=X+S:C=4:GOTO 410
370 Y=Y+S          :C=5:GOTO 410
380 Y=Y+S:X=X-S:C=6:GOTO 410
390       X=X-S:C=7:GOTO 410
400 Y=Y-S:X=X-S:C=8:GOTO 410
410 '
420 SOUND 0,C*5
430 SOUND 1,C
440 SOUND 8,10
450 SOUND 7,246
460 GOTO 250
470 '
480 DATA 1,3,7,15,14,12,12
490 DATA 14,15,31,31,63,127
500 DATA 127,3,7,128,192,224
510 DATA 240,112,48,48,112
520 DATA 240,248,248,252
530 DATA 254,254,192,224
```

①準備の部分（100〜220）
②スプライトパターン表示の部分（250）
③移動の部分（260〜400）
④音出力の部分（410〜460）
⑤データの部分（480〜530）

プログラムをBASICで書く前に，ロケットのデザインを決めなければなりません．デザインを決めたら，ロケットのパターンを1と0のデータに直し，スプライトパターンとして登録しておきます．

0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0
0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	0	1	1	1	0	0	1	1	1	0	0	0	0
0	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0
0	0	0	0	1	1	0	0	0	0	1	1	0	0	0	0
0	0	0	0	1	1	1	0	0	1	1	1	0	0	0	0
0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	1	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1	1	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0
0	0	0	0	0	0	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	1	0	0	0	0	0

ロケットのデザインパターン

データを作る手順は前にやったものと同じです．

リスト

```
100 '
110 DIM D(32)
120 FOR I=1 TO 16
130  INPUT A$
140  C$="&B"+LEFT$(A$,8)
150  D(I)=VAL(C$)
160  C$="&B"+RIGHT$(A$,8)
170  D(I+16)=VAL(C$)
180 NEXT I
190 '
200 FOR I=1 TO 32
210  PRINT D(I),
220 NEXT I
```

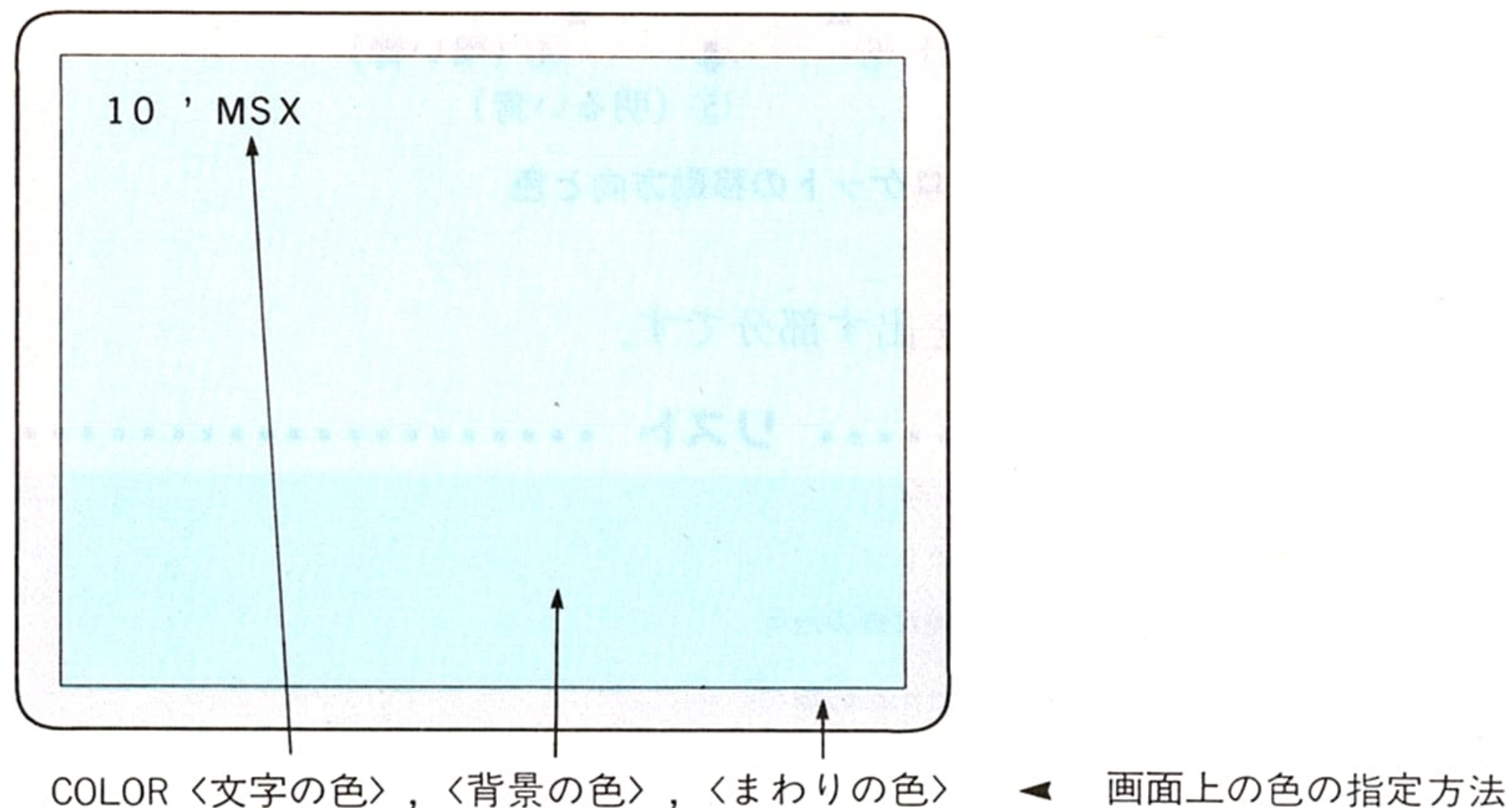
パターンができたら，いよいよプログラムを作っていきます．まずは準備の部分です．この部分は大きく分けて，①画面の準備，②スプライトパターンの登録，③音を使う準備，の3つの仕事があります．

① 画面の準備

100行で画面の準備を行っています．SCREEN 1，3 で16×16のスプライトパターンを拡大して使うことを指定しています．同じ行のCOLOR 1，2 は，画面の色を決めています．これまで円やスプライトパターンの色を指定したことはあっても，画面に表示される文字や，背景の色を実際に変えたことはありませんでした．前にも少し触れましたが，COLORを使うと，これらを変えることができます．

10 ' MSX

COLOR 〈文字の色〉，〈背景の色〉，〈まわりの色〉

◄ 画面上の色の指定方法

ここでは文字の色を 1 (黒)，背景の色を 2 (緑)にしています．

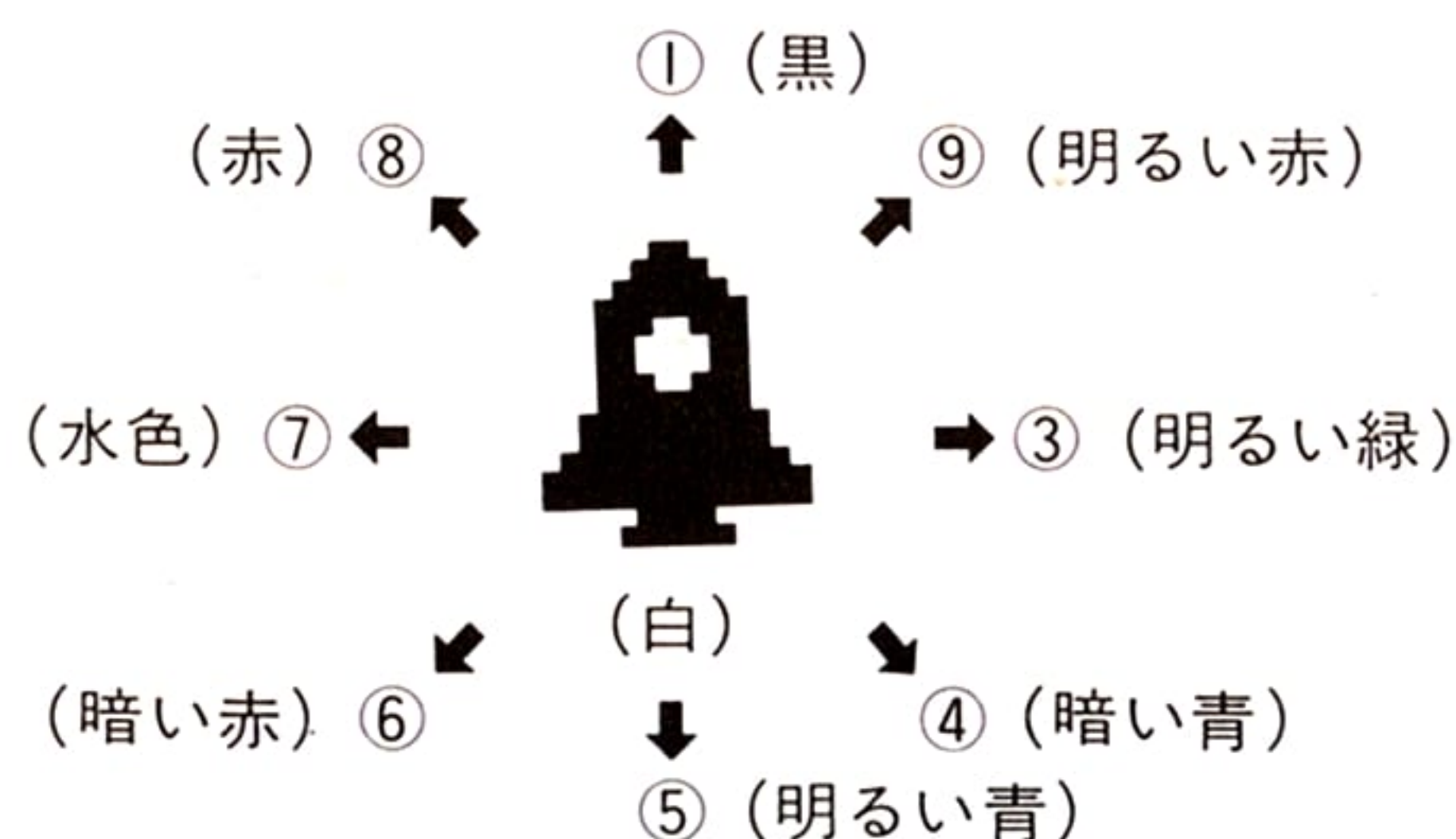
② スプライトパターンの登録

130～170行がスプライトパターンの登録の部分です．ここは，前にやった時と同じパターンです．

③ 音を使う準備

ここで，SOUNDが出てきます．180～200行の音を使うパターンを覚えて，これからもソックリ真似してください．

以上の準備ができたら，プログラムのメインになる部分に入ります．カーソル移動キーの方向と，ロケットの色は，次のように決めました．カーソル移動キーが押されてい

ないときは，乱数で上下左右にフワフワと動かします．カーソル移動キーの押され方に合わせて，実際にロケットを動かしている部分は，330～400行です．

ロケットの移動方向と色

さて，そのあとがいよいよ音を出す部分です．

リスト

```
410 '
420 SOUND 0,C*5   } 音の周波数の指定
430 SOUND 1,C     }
440 SOUND 8,10    } ボリュームの指定
450 SOUND 7,246   } 音の出力
```

420行，430行の2つで，どのような音にするのか決めています．SOUNDは，

SOUND　数字　，データ

の型で使います．数字の所には0，1，8，7の4つの数字を使います．

SOUNDによる音の指定方法

周波数の指定	SOUND 0，	周波数データ1（0～255）
	SOUND 1，	周波数データ2（0～15）
		★2つのデータの組合せで周波数が決まる
音量の指定	SOUND 8，	音量（0～15）
音の種類の指定	SOUND 7，	音の種類　246：ノイズの混じった音 254：澄んだ音 255：無音

```
SOUND 0,CC
SOUND 1,CC
```

上の2つをペアにして音色を変えることができます．CCの部分を変えるといろいろな音になります．

具体的にCCを変えるとどのような音になるか，次にサンプルを載せておきます．

リスト

```
100 '
110 C1=INT(RND(1)*100)
120 F=INT(RND(1)*2)
130 IF F=0 THEN CC=254
140 IF F=1 THEN CC=246
150 SOUND 0,C1
160 SOUND 1,0
170 SOUND 8,10
180 SOUND 7,CC
190 IF INKEY$<>"" THEN SOUND 7,255:END
200 GOTO 110
```

上のプログラムではC1の値が乱数を使って設定されるので，さまざまな音色の音が出ます．次にSOUND 8,10でボリュームを設定しています．ボリュームは0〜15の間の数字で指定し，0で無音，数字が大きくなればなるほど大きな音がします．

そして最後のSOUND 7,CCで実際に音を出します．CCの部分を変えることによって，澄んだ音にしたりノイズ(雑音)が入った音にしたりできます．CCが255のときは無音，254のときは澄んだ音，246のときはノイズが混じった音になります．このプログラムは何かキーを押すことで止まります．

SOUNDの0，1，8，7の4つを変えることによって，PLAYでは出せない音を創り出すことができます．

もっとくわしくSOUNDのことを知りたい人は，MSXを買うと付いてくるMSX BASICの文法書をよく読んでください．この例では，3つある音のチャンネルのうち1つのチャンネルしか使っていないのです．音を3つとも使うことで，さらに，迫力のある音楽や効果音を楽しむことができるでしょう．

COLUMN

色いろいろ

ＭＳＸでは16色のカラーを使うことができるのは，すでに見たとおりです．このように手軽にカラーグラフィックスが楽しめるのはよいのですが，ＣＯＬＯＲ命令を使っていると，まれに画面に文字が映らなくなってしまうことがあります．

たとえばどんな場合が考えられるでしょうか．電源を入れた状態ではＭＳＸは，背景の色が４（青）になっていますが，ここでうっかりｃｏｌｏｒ　４としてしまったような場合です．文字の色と，背景の色を同じにしてしまうと，当然文字が見えなくなってしまうのでこのようなことが起こります．

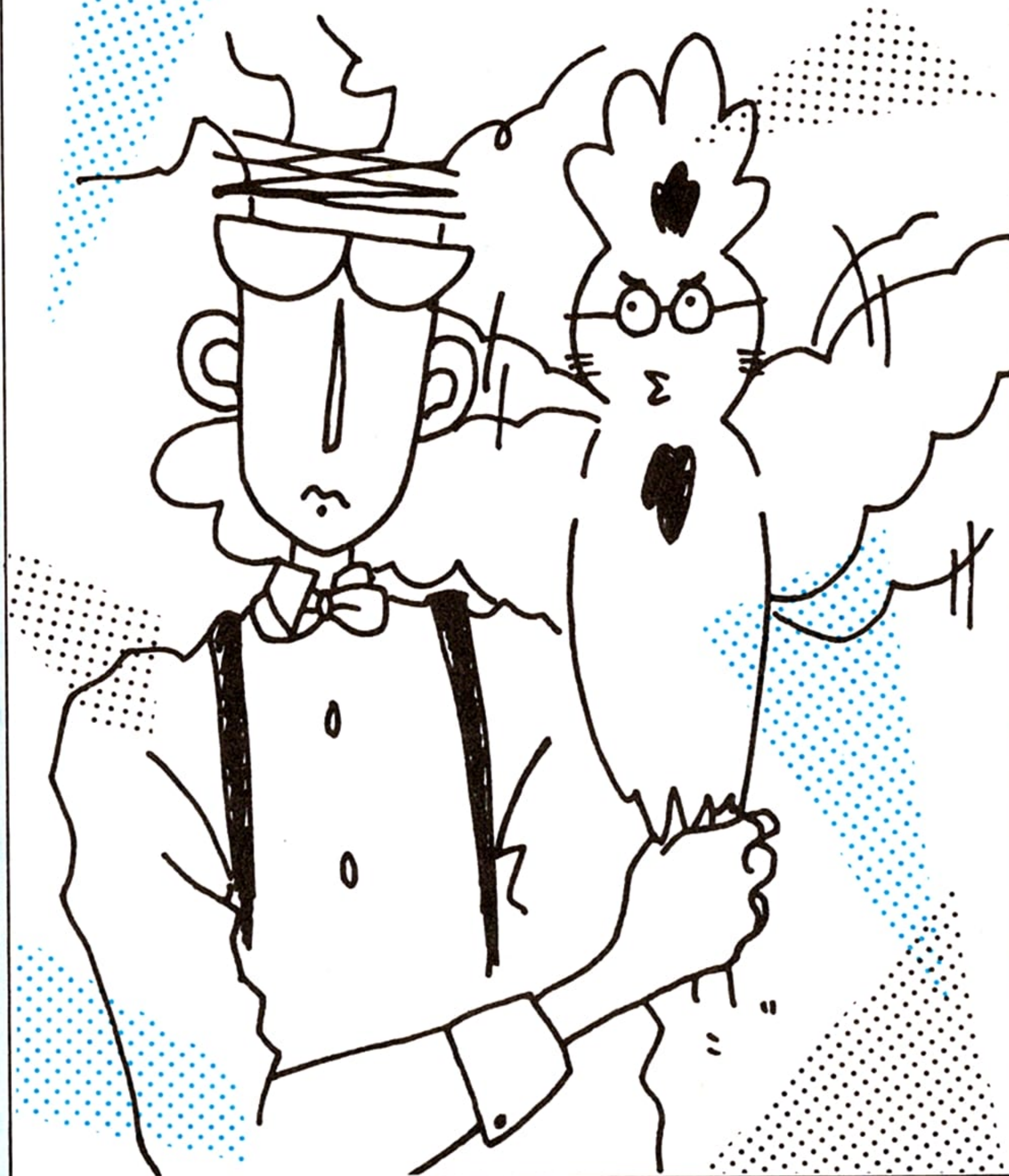
タイプに自信がある人は，画面になにも表示されなくてもｃｏｌｏｒ　15，７，４RETURNとすればもとに戻ります．自信のない人は電源を入れ直せばよいのですが，長いプログラムを打ち込んであったりする場合はそうもいきません．

このようなとき，覚えておくと便利なのが，ｆ・６キーの存在です．SHIFTキーを押しながらｆ・１キーを押してください．パッと画面が戻って文字が再び表れてきます．

このｆ・６キーには，ｃｏｌｏｒ　15，７，４　RETURNが登録されているので，このキーを１回押すと，画面がもとに戻るようになっているのです．

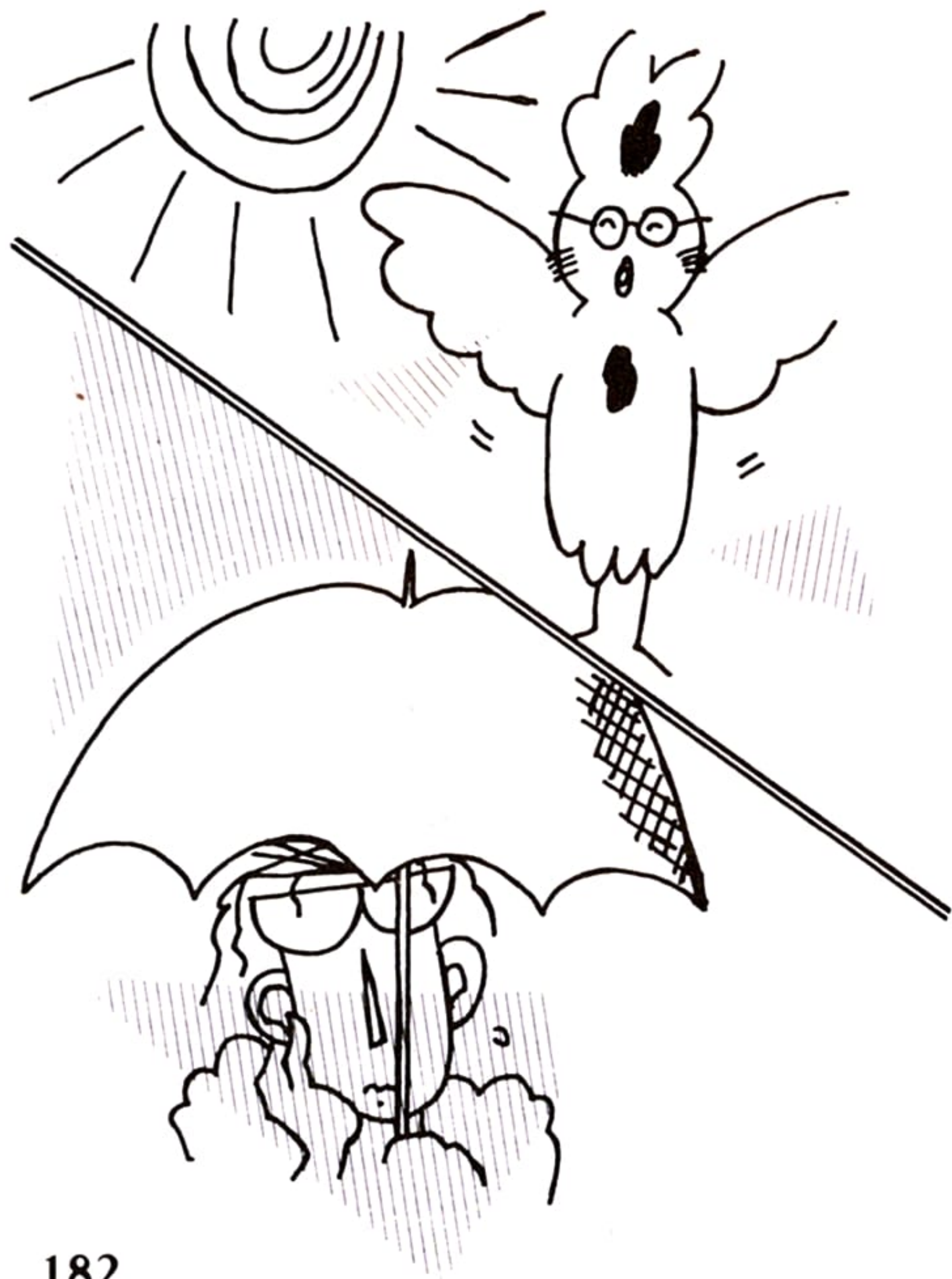
CHAPTER 5

これからが本格派

——バイオリズム・プログラムを作る——

MSX BASIC 5-1 プログラムを設計しよう

いよいよ最後のCHAPTERになりました．初めは恐る恐るだった人でも，今ではMSXをごく身近なものとして感じることができるようになったと思います．さて，あなたは今までこの本を読み，MSXに触れて，どんな知識を得たでしょうか．少しふりかえってまとめてみましょう．

○ コンピュータに何かをさせることの意味がわかった
○ プログラムというものの，おおまかな感じがつかめた
○ プログラムを作る上で必要な命令をいくつか覚えた

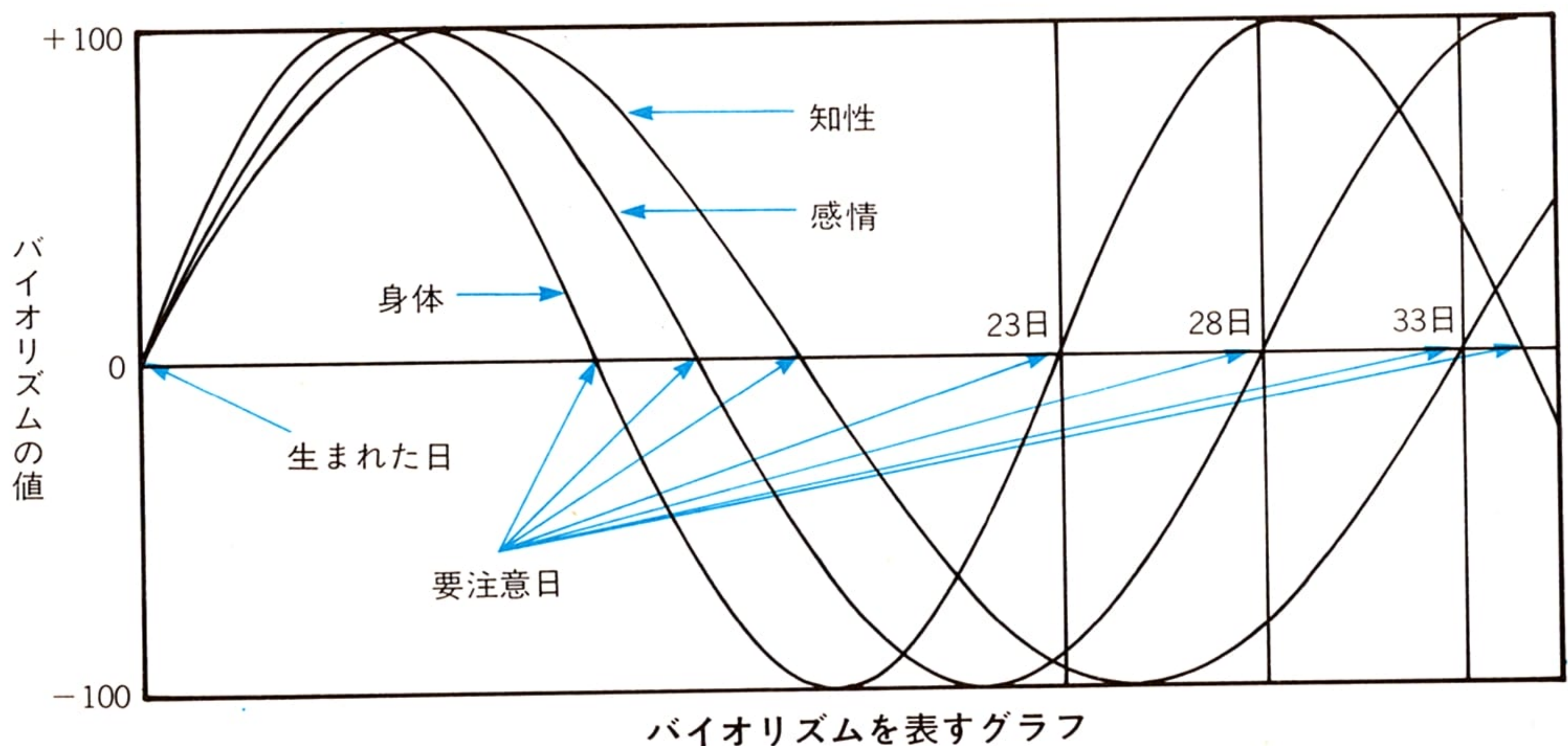
大体の人はこのような点に要約されると思います．

これらの知識をもとに，まだ知らない命令を使ったり，今までに作ったプログラムを手直ししたりすることは大変有益なことです．しかしMSXを使って，もっと高度な長いプログラムを作ろうと思うとき，これまでに得たあなたの知識では足りない何かを感じるのではないでしょうか．

このCHAPTERでは，バイオリズムのプログラムを例にとって，プログラムを作るときに最も重要となる考え方，いわばプログラム作りのセンスといったものに触れていきます．これからあなたが，自分の発想した機能をもつプログラムを作ろうと思ったとき，あるい

は作ったプログラムが使いづらかったり，予定していた機能が満たされないままあきらめてしまいそうになったとき，きっと役に立つ大切な考え方です．

バイオリズムとは

みなさんは〝バイオリズム〟という言葉を聞いたことがあると思います．耳にするのが初めてという人は，まず下の図を見てください．

バイオリズムを表すグラフ

人間は誰でも身体，感情，知性に，生まれた日から一定の周期で好不調が表れると言われ，その周期はそれぞれ23，28，33日となっています．バイオリズムの値はサインカーブと呼ばれる曲線を描いて，0を中心にプラス100からマイナス100の間で変化します．

それではこのグラフを見て，どのように判断すればよいのでしょうか．カーブがプラス側からマイナス側，またはその逆向きに，ちょうど0の値を横切るところに要注意日と書かれていますね．この日は，「不安定なので注意した方がよい」という意味なのです．

バイオリズムは占いなどと違い，人間が生物として固有の周期を持ち，その中で体調や感情などが不安定な時があるということを示してくれるものなのです．バイオリズム自体のしくみや理論に関して興味がある人は，バイオリズムに関する本が各種出版されているので読んでみるとよいでしょう（講談社ブルーバックスシリーズ『バイオリズムとは何か』など）．ここではバイオリズムの計算方法だけまとめてみることにします．

ある日のバイオリズムの値の求め方

100 sin（2πDA／CY）

CY → 身体，感情，知性のうち，求めたいものの周期を入れる．身体なら23，感情なら28，知性なら33を代入する

DA → 生まれた日から調べたい日までの間の総日数を計算し，さらにその値を求めたいものの周期で割った余りを代入する

まず考えることは

それではさっそく，バイオリズムをプログラムすることを考えてみましょう．

少し長いプログラム，少し高度なプログラムを作るときは，いきあたりばったりで命令を書いていってもすぐに行き詰まってしまい，そのうち自分が何をしているのかさえ見失ってしまいます．

今までの各CHAPTERでも何度か繰り返してきましたが，プログラムを作る上で最も大切なことは，作りたいものの内容を熟考し，その手順を決めていくことです．そして，このことこそ，プログラム作りの上で最も難しい部分であり，プログラムの善し悪しを決定してしまう部分なのです．

ではいったい，手順を決めるということはどういうことなのでしょうか．改めて考えてみます．

① そのプログラムの目的は何なのか，最終的にどういう結果が得られればよいのかイメージする

② その目的を満足させるために必要なデータは何かを整理する

③ 目的を満たすためには，どんな処理が必要か大まかに分けて考える

④ コンピュータに処理をさせる上で，それぞれの処理が混乱しないよう整理する

では，バイオリズムのプログラムの場合，具体的にはどのようになるかを考えていきましょう．

まず目的についてです．

このプログラムでは最終的に，バイオリズムを調べたい月のバイオリズムカーブを，

さきほどの図のように画面に描くことにします．

次に，必要なデータです．

このプログラムでは，(1)調べたい人の生年月日，(2)調べたい日の年月日，のデータが必要なので，MSXの画面からの指示に従って，キーボードから入力することにします．

次に，どんな処理が必要かを考えてみます．

① プログラムの初めの状態を設定する部分
② 日付を入力する部分
③ 総日数を求める部分
④ バイオリズムの値を計算し，画面に表示させる部分
⑤ 終了処理のための部分

が考えられるでしょう．

これらのそれぞれの処理については，この次のsectionから詳しく考えることにします．

このように，プログラムを作る手順のうち特に重要な点は，処理を大まかな部分に分けて考えていく，ということです．その理由はいくつか考えられますが，主に次のようなメリットが挙げられるでしょう．

○ 大きなプログラムでも，小さな部品の集まりとして考えると，設計の見通しがつきやすい
○ 間違えた部分を発見しやすくなり，修正(デバッグ)が楽
○ 各部分ごとに置き換えられるので，あとで変更を加えることが簡単
○ 構成がわかりやすいので，プログラム作成後時間が経っても，変更を行いやすい
○ 巨大なプログラムを作るときは，各ブロックごとに分けて，複数の人で共同してプログラムを作ることができる

この他にもたくさんのメリットがあるのですが，具体的にならないと理解しづらいと思います．

これからの４つのsectionでは，ここに書いた考え方を，バイオリズムのプログラムを作っていく過程で，さらに具体的に説明していくことにしましょう．

MSX BASIC

5-2 はじめを大切に

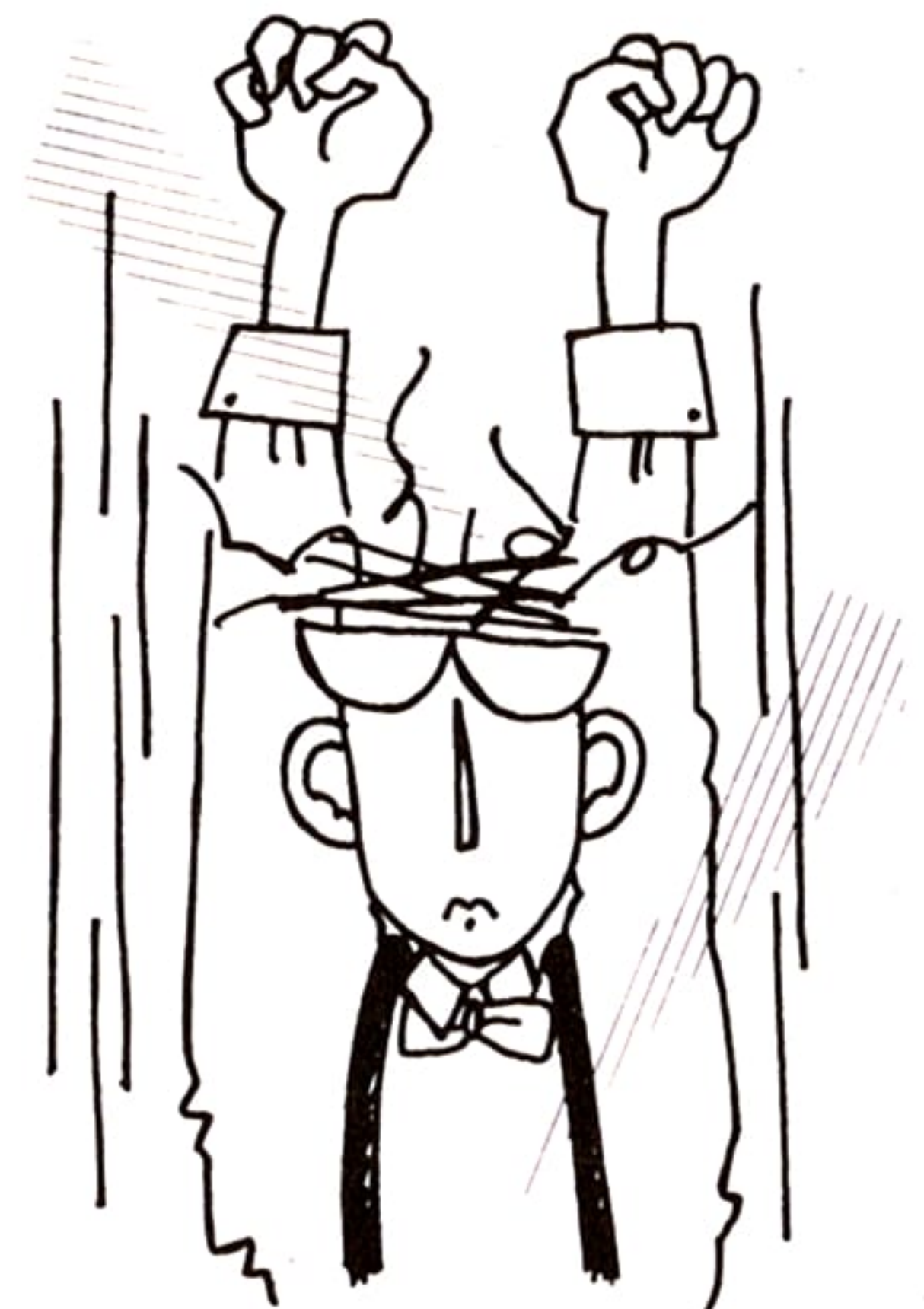
プログラムの大まかな作り方がわかったところで，それぞれの処理のプログラムを作っていきましょう．だいたいの構造はすでに前のsectionで説明したとおりです．これをさらに，具体的に整理して，プログラム作りの細かなテクニックを紹介していきます．

わかりやすいプログラム

大きなプログラムを作るときには，プログラムをわかりやすく書く必要があります．10行や20行で終わるプログラムなら，リストを追うことで，何をしているのかだいたいの見当がつきます．でも，それが100行を超えてしまうと，自分で作ったものでもわからなくなってしまうのが普通です．

プログラムをわかりやすく，見やすくするための方法のひとつに，コメント(注釈)があります．次のリストを見てください．

リスト

```
10 '**************************
20 '***                    ***
30 '***  ﾊﾞｲｵﾘｽﾞﾑ ﾌﾟﾛｸﾞﾗﾑ  ***
40 '***                    ***
50 '**************************
```

これは，これから作るプログラムの先頭の部分です．アポストロフィ[’]に続く文字は，前に説明したとおりプログラムの実行には影響ありません．ここではプログラムのタイトルを書いて，何のプログラムであるかを明示しています．これと同じように各処理の名称もコメントを使って書けば，プログラム全体の構造がよくわかるようになるでしょう．

プログラムの構造をわかりやすくする方法は，他にもあります．次のリストを見てください．

リスト

```
1000 '
1010 '************ メイン ルーチン
1020 '
1030 GOSUB 5000:'======> しょきせってい へ
1040 GOSUB 6000:'======> しょきか゛めん へ
1050 GOSUB 10000:'=====> にゅうりょく ルーチン へ
1060 GOSUB 20000:'=====> にっすう へんかん ルーチン へ
1070 GOSUB 30000:'=====> しゅうき あまりけいさん ルーチン へ
1080 GOSUB 40000:'=====> ハ゛イオ ひょうし゛ ルーチン へ
1090 GOSUB 50000:'=====> けいそ゛く にゅうりょく ルーチン へ
1100 IF CO=1 THEN 1040
1110 END
```

CHAPTER 4で説明したGOSUB命令が並んでいますね．このプログラムでは，さきに機能分けしたそれぞれの処理をサブルーチンの形でまとめておいて，先頭で呼び出しているのです．このようにすると，プログラムの最初の何行かを見るだけで全体の構造がよくわかります．

このように全体の流れをまとめた部分を，メインルーチンと呼んでいます．このリストのように，各サブルーチンを呼び出す命令のあとにコメントでサブルーチンの働きを書いておくと，さらにわかりやすくなります．なお，前のsectionの機能分類と少し違うところは，プログラムの構造を熟慮した結果と考えてください．

準備(初期設定)の部分

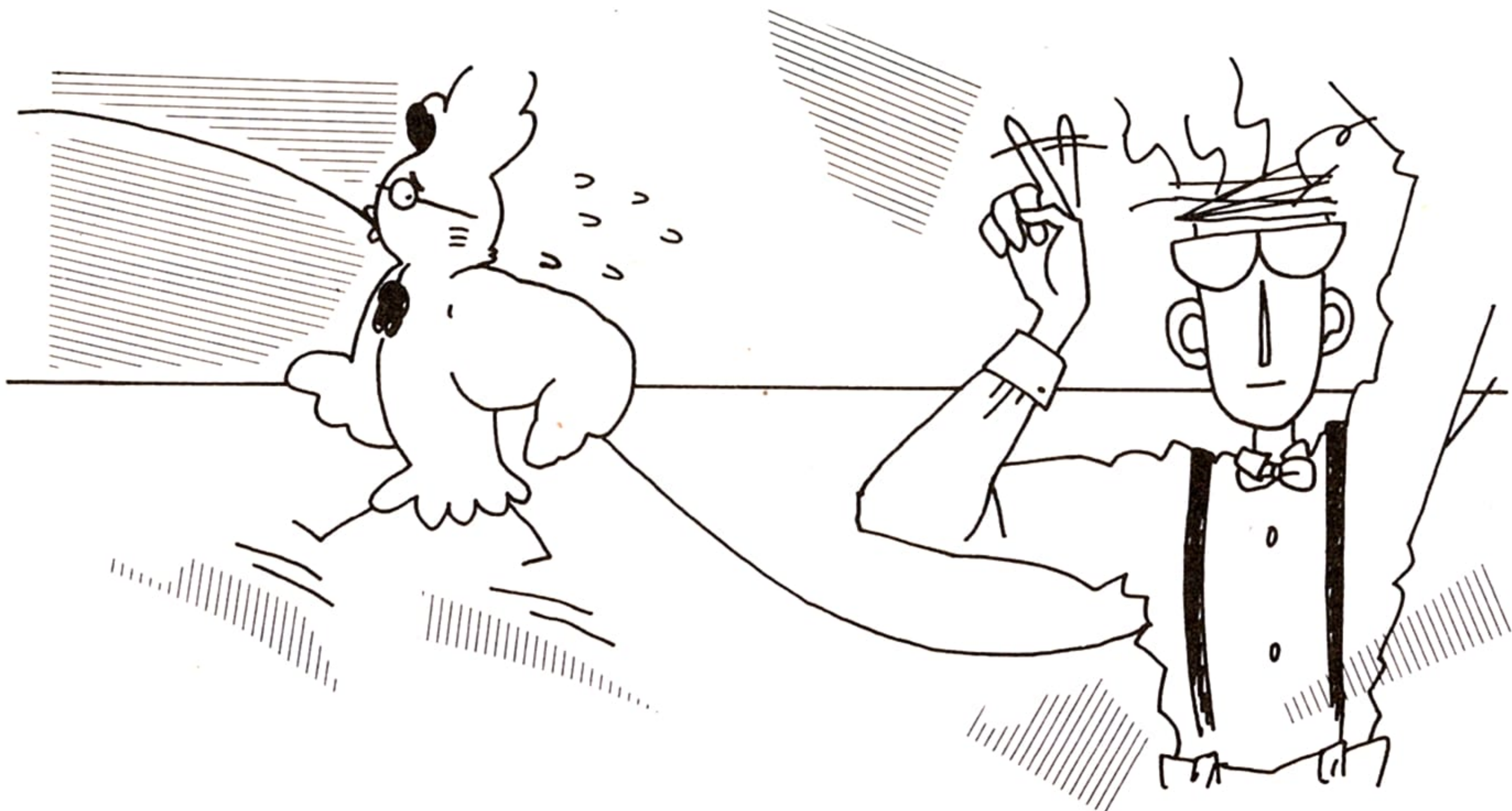
プログラム全体の構造の見通しがついたら，実際にプログラムを作っていくことにしましょう．まずは準備の部分です．この部分は行番号の5000番台を使うことにします．

準備といってもいったい何をすればよいのでしょうか．今までの例から考えて，画面の設定と，変数の初期値の設定，配列の宣言と初期値の設定，といったことをする必要があるようです．

まず画面の設定をします．バイオリズムのカーブを描く前に，誕生日やバイオリズムを調べたい日を入力するので，文字を表示させる画面がよいでしょう．色は文字が白，まわりと背景は黒にしておくと見やすいと思います．以上のことをプログラムにすると次のようになります．

リスト

```
5000 '
5010 '****************** しょきせってい
5020 '
5030 SCREEN 1
5040 COLOR 15,1,1:CLS
```

準備の部分で配列の宣言なども行います．宣言された配列の使いみちは，次に示すとおりです．

M(12)…………1～12月の1日が1年の最初から何日目かを表す配列
DA(3)………バイオリズムを書くときに使う
CY(3)………身体，感情，知性の周期
COL(3)………身体，感情，知性のカーブの色

この部分のリストは，次のようになります．

リスト

```
5050 DIM M(12),DA(3),CY(3),COL(3)
5060 FOR I=1 TO 12
5070  READ M(I)
5080 NEXT I
5090 FOR I=1 TO 3
5100  READ CY(I)
5110 NEXT I
5120 FOR I=1 TO 3
5130  READ COL(I)
5140 NEXT I
5150 RETURN
```

5050行で配列の宣言をしています．いくつもの配列をカンマ[，]で区切ることによって同時に宣言できます．5060～5140行では，宣言した配列の要素に具体的な値を読み込んでいきます．このデータは60000行～60050行で用意します．

リスト

```
60000 '
60010 '********** で゛ーた
60020 '
60030 DATA 0,31,59,90,120,151,181,212,243,273,304,334
60040 DATA 23,28,33
60050 DATA 13,2,7
```

これで，準備の部分は終わり，5150行でメインルーチンに戻り，次の作業へと進んでいきます．

MSX BASIC

5-3 日にちのチェックと日数計算

さて，いよいよ誕生日と，バイオリズムを調べたい日を入力して，日数計算のあとバイオリズムの値を求める部分を作っていきましょう．機能別に説明を続けていきます．

初期画面の表示

バイオリズムデータを入れる前に，何のプログラムであるかを示す画面を描かせておきましょう．デザインは手間がかからず，見映えのするものにしました．この部分は6000番台を使います．

このようなタイトル画面が表示されたあとで，計算に必要なデータを入力することにします．

リスト

```
6000 '
6010 '********** しょきか゛めん
6020 '
6030 PRINT " ***************************"
6040 PRINT " ***                     ***"
6050 PRINT " ***  ハ゛イオリス゛ム フ゜ロク゛ラム  ***"
6060 PRINT " ***                     ***"
6070 PRINT " *** あなた の コンテ゛ィション は ? ***"
6080 PRINT " ***                     ***"
6090 PRINT " ***************************"
6100 RETURN
```

年月日の入力

年月日の入力をするためにはINPUT命令を使えばよいでしょう．誕生日の西暦をY1，月をM1，日をD1と決めて，次のようにすれば，とりあえず目的は満足します．

```
10000  INPUT "年"; Y1
10010  INPUT "月"; M1
10020  INPUT "日"; D1
```

```
   012                        27
0  ****************************
   ***                      ***
   ***   バ゛イオリズ゛ム プ゜ログ゛ラム   ***
   ***                      ***
   *** あなた の コンテ゛ィション は ? ***
   ***                      ***
6  ****************************

8   あなたの うまれは せいれき なん年 ?

10  なん月 ?

12  なん日 ?
```

でも，これだけでは入力画面のレイアウトに不満が残ります．そこでLOCATE命令を使い，位置を指定するようにしましょう．

ところで，INPUT命令で数値データを入力するようにしておくと，間違って他の文字を入れてしまったときに，Redo from startというメッセージが出ることを知っていますか．このメッセージが出てしまうとせっかくレイアウトした画面が崩れてしまいます．これを防ぐためには，さらに工夫が必要です．

INPUT命令で，文字変数にデータを入力するようにしておけば，入力するデータがなんであれRedo from startのメッセージは出てきません．そこで，年，月，日などの数字を数値としてではなく，文字として入力するようにします．もちろん文字のままでは，計算することができないので，これを数値に変換することが必要です．VAL命令を思い出してください．これを使えば文字(数字)を数値に変換できます．

```
INPUT  A$
Y1=VAL(A$)
```

このようにすれば，Redo from start が出るのを防ぐことができます．

入力部分で絶対に考慮しなくてはならない問題として，入力チェックがあります．年がマイナスになっていないか，月が1～12の範囲に収まっているか，日は1～31の間になっているか，などをチェックしなければなりません．1985年13月62日などというのは一体いつのことかわかりませんね．このチェックのためにはIF～THENを使います．

年，月，日のそれぞれについて数値をチェックして，おかしな値だったらもう一度入力します．このとき，前に入力した数字などが画面上に残っていると再入力しづらいので，前の入力で残っている表示は消しておきます．

以上のような工夫の結果，誕生日と，バイオリズムを調べたい日（Y2，M2，D2）の年月日を入力する部分のリストは，次のようになりました．

リスト

```
10000 ´
10010 ´********** にゅうりょく ルーチン
10020 ´
10030 ´----------- うまれた 日
10040 A$=""
10050 LOCATE 2,8:PRINT SPC(26)
10060 LOCATE 2,8:INPUT "あなた の うまれは せいれき なん年 ";A$
10070 Y1=VAL(A$)
10080 IF Y1=<0 THEN 10050
10090 LOCATE 2,10:PRINT SPC(26)
10100 LOCATE 2,10:INPUT"なん月 ";A$
10110 M1=VAL(A$)
10120 IF M1<1 OR M1>12 THEN 10090
10130 LOCATE 2,12:PRINT SPC(26)
10140 LOCATE 2,12:INPUT"なん日 ";A$
10150 D1=VAL(A$)
10160 IF D1<1 OR D1>31 THEN 10130
10170 LOCATE 0,8
10180 FOR I=1 TO 12
10190  PRINT SPC(28)
10200 NEXT I
12000 ´----------- しらべる 日
12010 A$=""
12020 LOCATE 2,8:PRINT SPC(26)
12030 LOCATE 2,8:INPUT "しらべたい 日 は せいれき なん年 ";A$
12040 Y2=VAL(A$)
12050 IF Y2=<0 THEN 12020
12060 LOCATE 2,10:PRINT SPC(26)
12070 LOCATE 2,10:INPUT"なん月 ";A$
12080 M2=VAL(A$)
12090 IF M2<1 OR M2>12 THEN 12060
12100 LOCATE 2,12:PRINT SPC(26)
12110 LOCATE 2,12:INPUT"なん日 ";A$
12120 D2=VAL(A$)
12130 IF D2<1 OR D2>31 THEN 12100
12140 RETURN
```

10050行で今から文字を書くところをきれいにしています．SPC(26)というのは26個の空白を表すので，PRINT SPC(26)は26文字分の空白を表示することになります．

10060～10160行で，年月日それぞれの入力とチェックを行い，10170～10200行で，次に備えて画面を掃除します．調べたい日の入力も，考え方は同じです．

日数計算

バイオリズムの値を求めるためには，生まれてから何日たっているのかを計算して，それを身体，感情，知性の周期で加工していきます．

リスト

```
20000 '
20010 '********** にっすう に へんかん
20020 '
20030 YY=Y1:MM=M1:DD=D1
20040 GOSUB 65000:'======> にっすう へんかん へ
20050 D1=DA
20060 '---------- せいそ゛ん にっすう
20070 YY=Y2:MM=M2:DD=D2
20080 GOSUB 65000:'======> にっすう へんかん へ
20090 DI=DA-D1
20100 '---------- ハ゛イオち の けいさん
20110 DA=DI-DD
20120 RETURN
30000 '
30010 '********** しゅうき あまりけいさん ルーチン
30020 '
30030 FOR I=1 TO 3
30040  DA(I)=DA-CY(I)*INT(DA/CY(I))
30050 NEXT I
30060 RETURN

65000 '
65010 '********** にっすう けいさん うるう年 ほせい
65020 '
65030 DA=YY*365+M(MM)+DD
65040 DA=DA+INT(((YY-1)/4)+1)
65050 DA=DA-INT(((YY-1)/100)+1)
65060 DA=DA+INT(((YY-1)/400)+1)
65070 IF (MM>2) AND (YY/4=INT(YY/4)) THEN DA=DA+1
65080 RETURN
```

紀元前1年の1月1日を第1日目として，調べたい日までの日数から誕生日までの日数を引くと，生まれてからの日数がわかります(DI)．次に，グラフの表示は月の初めから行うので，前月の最後の日が生後何日目かも求めます（DA)．周期余りの計算の部分では，このDAが各周期の始点から何日目なのかを求めています（DA(I))．

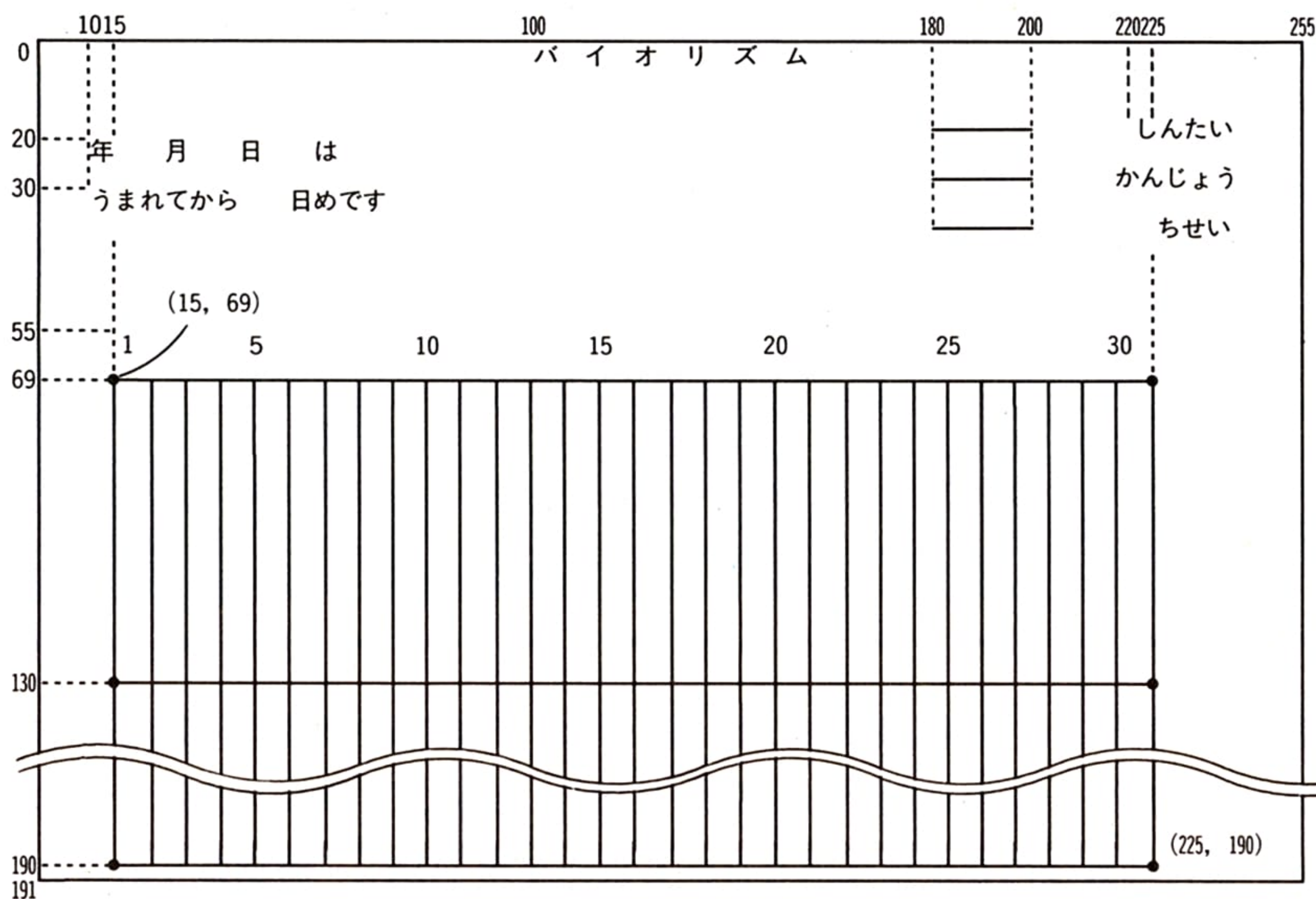
MSX BASIC

5-4 バイオリズムを描く

いよいよバイオリズムをグラフにしてみましょう．表示部分は大きく分けると，①タイトルなどを書く部分，②ワク組みを書く部分，③グラフを描く部分，の 3 つに分けることができます．まずバイオリズムを描く画面のレイアウトを決め，それに沿って①〜③を作っていくことにしましょう．

画面のレイアウト

テキスト画面ではグラフを描けないので，SCREEN2のグラフィック画面を使います．この画面にどのように表示するのか，そのレイアウトを作ります．

10 15 100 180 200 220 225 255
0
バ イ オ リ ズ ム
20 年 月 日 は
30 うまれてから 日めです
しんたい
かんじょう
ちせい
(15, 69)
55 1 5 10 15 20 25 30
69
130
190 (225, 190)
191

バイオリズムを表示する画面のレイアウト

タイトルなどの表示

ここでは，「バイオリズム」というタイトルと，調べたい日が生まれてから何日たっているかの表示，身体，感情，知性のカーブがそれぞれ何色なのかを示す凡例を描くことにします．

リスト

```
40000 '
40010 '********** ひょうし" と けいさん
40020 '
40030 SCREEN 2:COLOR 15,1,1:CLS
40040 OPEN "GRP:" FOR OUTPUT AS#1
40050 PRESET(100,0):PRINT#1,"ハ"イオリス"ム"
40060 PRESET(10,20):PRINT#1,Y2;"年";M2;"月";D2;"日は "
40070 PRESET(10,30):PRINT#1," うまれてから";DI;"日めて"す。"
40080 LINE(180,20)-(200,20),13
40090 PRESET(220,16):PRINT#1,"しんたい"
40100 LINE(180,30)-(200,30),2
40110 PRESET(205,26):PRINT#1,"かんし"ょう"
40120 LINE(180,40)-(200,40),7
40130 PRESET(230,36):PRINT#1,"ちせい"
```

グラフィック画面に字を書くにはさきに説明したとおり，OPEN "GRP:" FOR OUTPUT AS#1と，PRINT#1を使っています．文字の位置を指定する方法として，すでにPSET(X，Y)，4を使う方法を紹介していますが，今回はPRESET(X，Y)を使ってみました．これはもともとは点を消す命令ですが，これを使うと色の指定をしなくてもよいというメリットがあります．

バイオリズムのカーブを描くときに，何色がどのカーブかわかるような凡例もつけておきます．身体は紫，感情は緑，知性は水色にすると，それぞれの色で線を引きます．線を引く命令はLINEでしたね．

枠組みを描く

タイトル，凡例などを描いたら，グラフを描くまえに必要な枠や目盛を描きます．

1～31の31本の縦線を引き，外側の枠と，中心の線を引きます．このとき，5，10，15，……と5ずつ日を表示しましょう．

リスト

```
40140 '
40150 '----------- わくぐみ を かく
40160 '
40170 PRESET (15,55):PRINT#1,"1"
40180 FOR X=1 TO 31
40190  LINE(X*7+8,68)-(X*7+8,190),15
40200  IF X MOD 5=0 THEN PRESET(X*7-4,55):PRINT#1,X
40210 NEXT X
40220 LINE(15,130)-(225,130),15
40230 LINE(15,69)-(225,190),15,B
```

表示するのは5で割った余りが0になる日です．MSX BASICには余りを求める演算子，MODが用意されているので，それを使ってみました（40200行）．

```
X  MOD  5
```

の結果は，Xを5で割ったときの余りです．X＝1のときは1，X＝5のときは0，X＝10のときも0となります．

カーブを描く

枠が描けたら，いよいよバイオリズムのカーブを表示していきます．

リスト

```
40240 '
40250 '----------- かーぶ を かく
40260 '
40270 FOR I=1 TO 3
40280  PSET(15,130-60*SIN(2*3.14159*(DA(I))/CY(I)))
40290  FOR J=1 TO 31 STEP .5
40300   Y=130-60*SIN(2*3.14159*(J+DA(I))/CY(I))
40310   LINE -(J*7+8,Y),COL(I)
40320  NEXT J
40330 NEXT I
40340 RETURN
```

バイオリズムの身体，感情，知性のカーブはそれぞれサインカーブの一種です．三角関数が出てくると頭が痛くなる人は，ここは何も考えず打ち込んでもいいでしょう．

ただしSINの値のままでは，−1から1の間の値を取るので，これをそのまま座標上に表示させてもほとんど意味がありません．また普通，グラフの座標は真中が0で，上へいけば+，下へいけばーになっています．が，グラフィックモードの座標は左上隅が0で，下へいけば+になっているので，これも考慮する必要があります．

このような違いを調整して，さきに設計した画面の中にグラフが収まるように工夫しているのが40240〜40340行です．40280行で始点を定めて，40290〜40320行で，調べたい日を含む年月の1日から31日に対応するグラフを描いています．

終わりの部分

バイオリズムのカーブを描き終わったら，続けてバイオリズムを調べてみるのか，プログラムを終わらせるのかを決めます．

リスト

```
50000 '
50010 '************ けいぞ"く はんてい まち
50020 '
50030 CO=0
50040 PRESET(10,40)
50050 PRINT#1," ほか の 日 を しらべ"ますか ? "
50060 A$=INPUT$(1)
50070 IF A$="y" THEN CO=1
50080 CLOSE#1
50090 SCREEN 1:WIDTH 29
50100 RETURN
```

「ほかの日をしらべますか」の問に，yを押すと再びプログラムの最初に戻ります．ここで変数COがプログラムを続けるのか終わりにするのかのスイッチになっています(実際に終わりにするかどうかを決めているのはメインルーチンの1100行です)．

リスト

```
1090 GOSUB 50000:'=====> けいぞ"く にゅうりょく ルーチン へ
1100 IF CO=1 THEN 1040
1110 END
```

これでプログラムはできあがりです．次に実際に試してみることにしましょう．

MSX BASIC

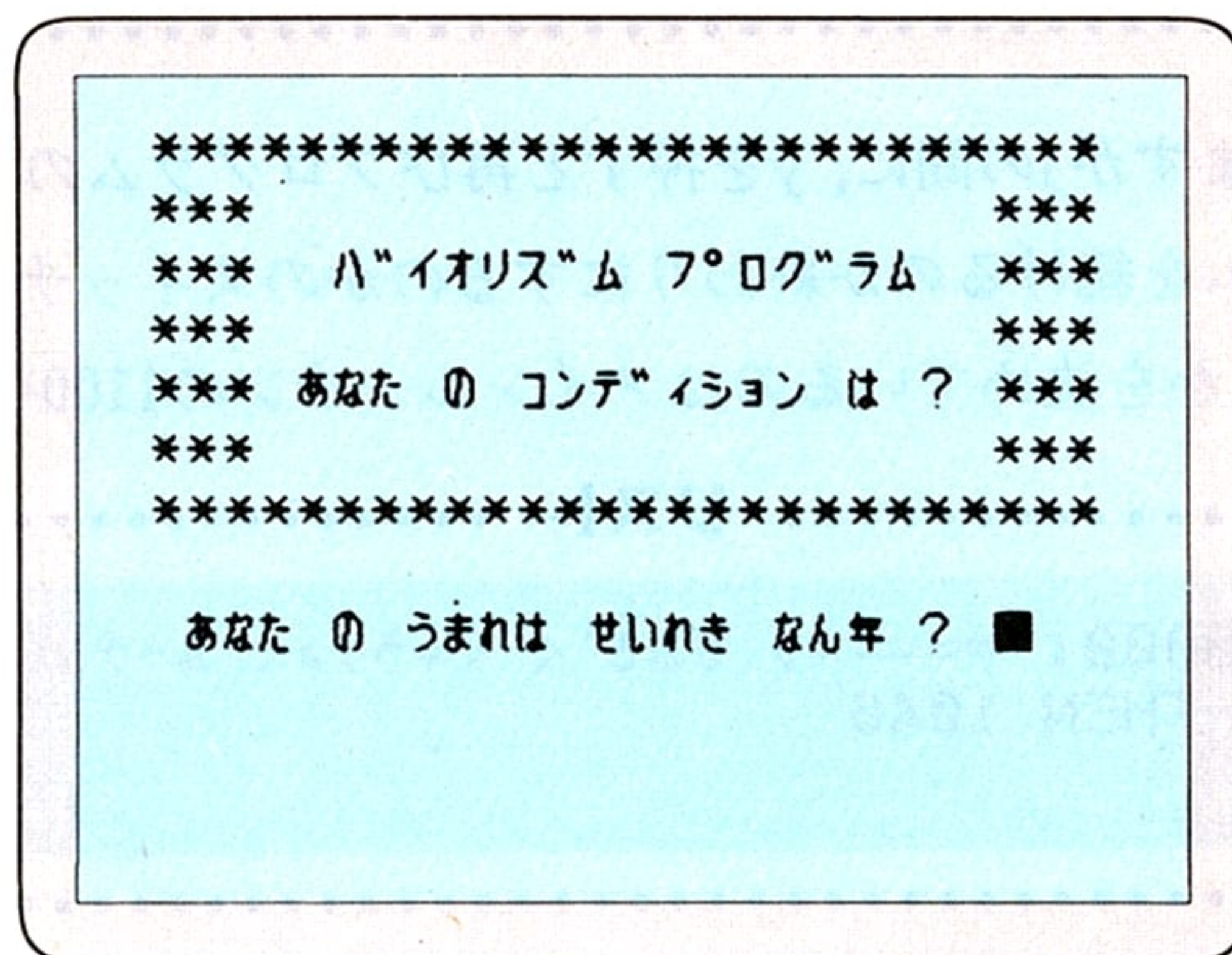
5-5 調べてみよう バイオリズム

この本の総仕上げとして作ったバイオリズムも，ひとまず完成しました．ちょっとリストを取ってみてください．部分部分を作っているうちはそれほど感じませんが，全部まとめてみるとかなり長い本格的なプログラムになってしまいました．一気にこれだけ大きなプログラムを作ろうとしてもイヤになってしまいますが，これまでしてきたように，プログラムを機能によって部分部分に分け，一つひとつの部分の手順を整理してプログラムを作れば，思ったより簡単にできてしまいます．

さあ，このプログラムを動かしてみましょう．

日にちの入力

[f・5] またはrun [RETURN] でプログラムを動かしてください．次のような画面が表れます．

```
***************************
***                     ***
***  ﾊﾞｲｵﾘｽﾞﾑ ﾌﾟﾛｸﾞﾗﾑ  ***
***                     ***
*** あなた の ｺﾝﾃﾞｨｼｮﾝ は ? ***
***                     ***
***************************

 あなた の うまれは せいれき なん年 ? ■
```

自分の生まれた年を西暦で入力します(1965年7月7日生まれの場合).

1965 RETURN

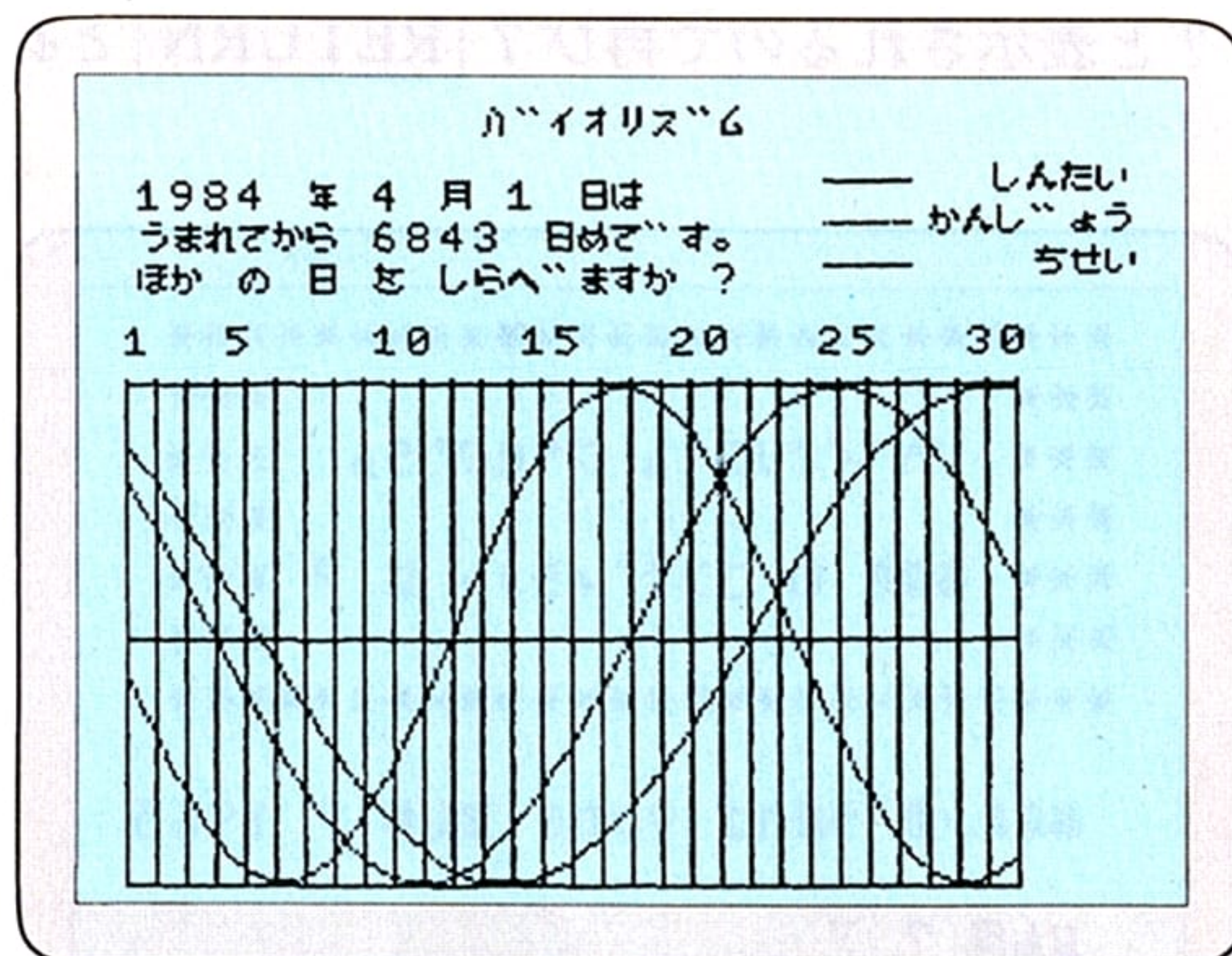
続いて，何月？と，月を入れるように求めてきますので，7 RETURN のように入力します．続いて，何日？と表示されるので再び7 RETURN と打ち込みます．

```
***************************
***                     ***
***  ハ゛イオリス゛ム フ゜ロク゛ラム  ***
***                     ***
*** あなた の コンテ゛ィション は ? ***
***                     ***
***************************

あなた の うまれは せいれき なん年 ? 1965

なん月 ? 7

なん日 ? 7■
```

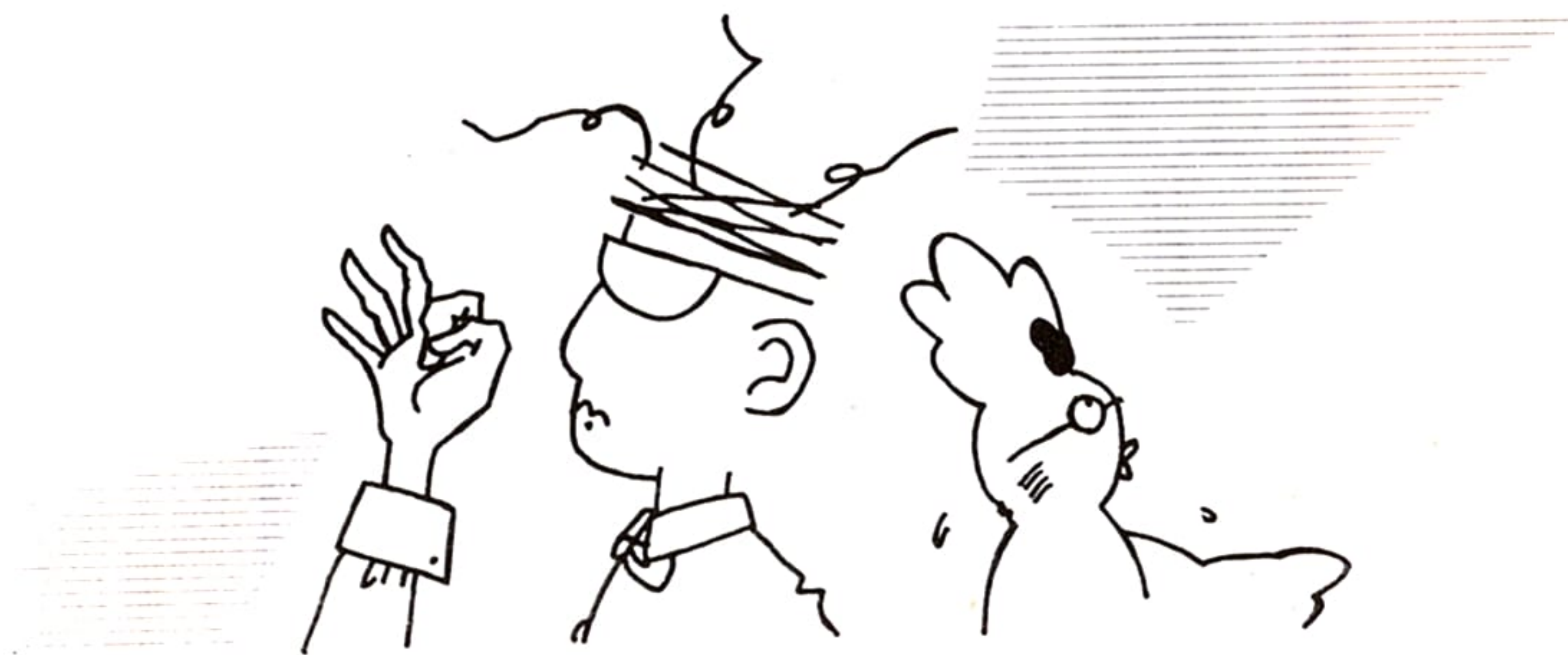
誕生日の入力が済んだら，続いてバイオリズムの値を表示させたい年月日を入力します．誕生日の入力の時と同じように年，月，日を画面の指示にしたがって入れていってください．

```
***************************
***                     ***
***  ハ゛イオリス゛ム フ゜ロク゛ラム  ***
***                     ***
*** あなた の コンテ゛ィション は ? ***
***                     ***
***************************

しらへ゛たい 日 は せいれき なん年 ? 1984

なん月 ? 4

なん日 ? 1■
```

バイオリズムの表示

日にちの入力を終了すると，画面が切り替わってバイオリズムのカーブが表示されます．

身体のカーブは紫，感情のカーブは緑，知性のカーブは水色で表示されています．カーブがゼロを横切るときは要注意日です．あらかじめバイオリズムを調べておいてコンディションには充分注意しましょう．

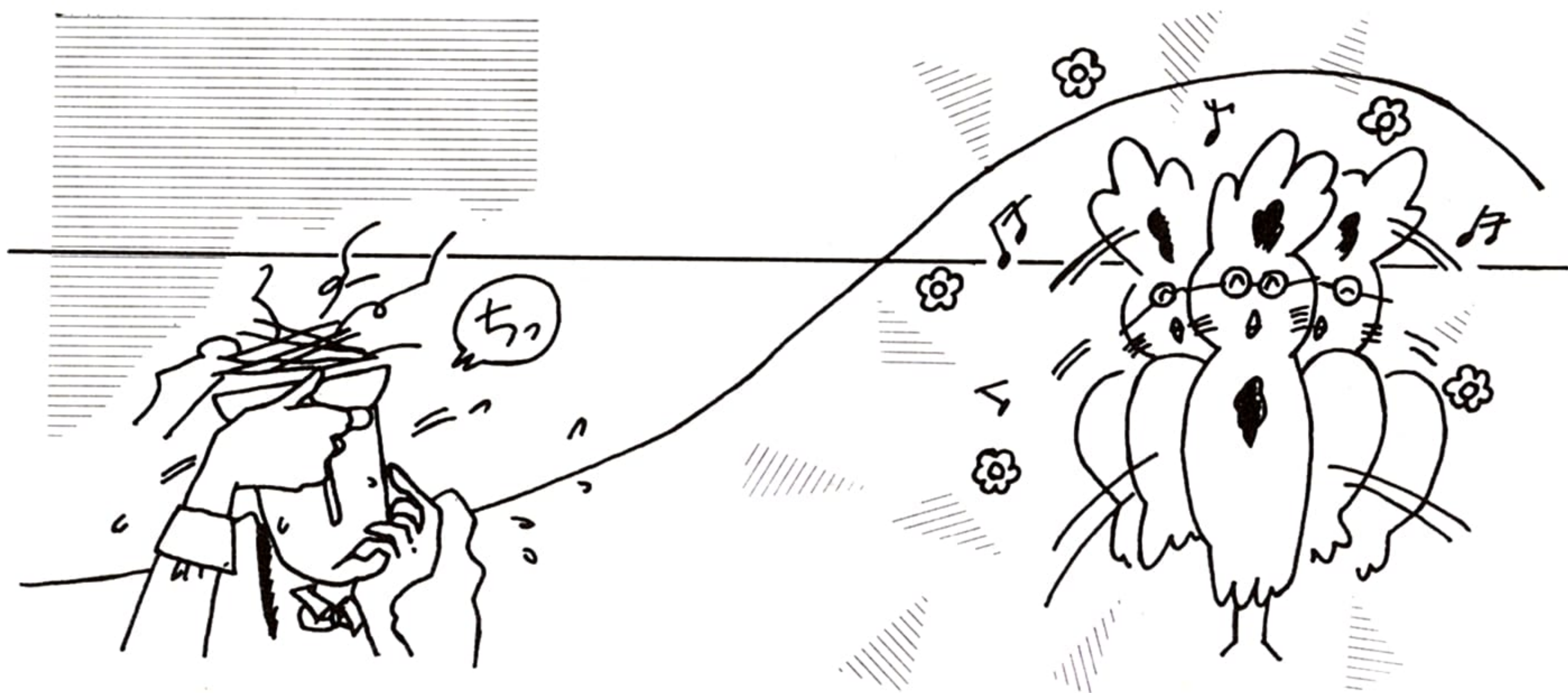
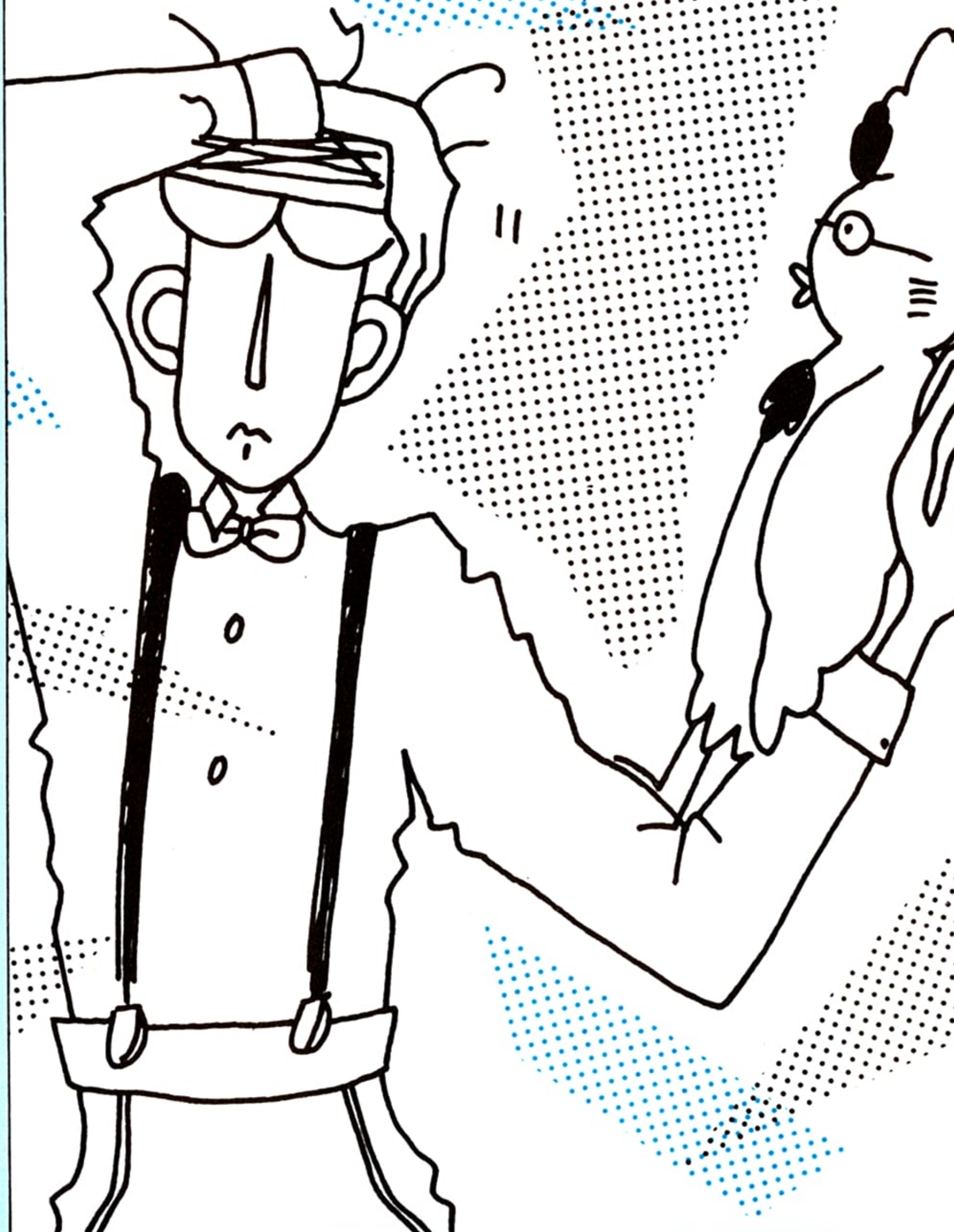
カーブを描き終わると，「ほか　の　日を　しらべますか？」とたずねてきます．ここでyを押すと最初に戻り，別のバイオリズムの計算をします．それ以外のキーを押すと，プログラムは終わります．

最後にこのプログラムの全リストを載せておきましょう．

リスト

```
10 '****************************
20 '***                      ***
30 '***  ハ゛イオリス゛ム フ゜ロク゛ラム  ***
40 '***                      ***
50 '****************************
1000 '
1010 '*********** メイン ルーチン
1020 '
1030 GOSUB 5000:'======> しょきせってい へ
1040 GOSUB 6000:'======> しょきか゛めん へ
1050 GOSUB 10000:'=====> にゅうりょく ルーチン へ
1060 GOSUB 20000:'=====> にっすう へんかん ルーチン へ
1070 GOSUB 30000:'=====> しゅうき あまりけいさん ルーチン へ
1080 GOSUB 40000:'=====> ハ゛イオ ひょうし゛ ルーチン へ
1090 GOSUB 50000:'=====> けいそ゛く にゅうりょく ルーチン へ
1100 IF CO=1 THEN 1040
1110 END
5000 '
5010 '***************** しょきせってい
5020 '
5030 SCREEN 1
5040 COLOR 15,1,1:CLS
5050 DIM M(12),DA(3),CY(3),COL(3)
5060 FOR I=1 TO 12
5070  READ M(I)
5080 NEXT I
5090 FOR I=1 TO 3
5100  READ CY(I)
5110 NEXT I
5120 FOR I=1 TO 3
5130  READ COL(I)
5140 NEXT I
5150 RETURN
6000 '
6010 '********** しょきか゛めん
6020 '
6030 PRINT " ****************************"
6040 PRINT " ***                      ***"
6050 PRINT " ***  ハ゛イオリス゛ム フ゜ロク゛ラム  ***"
6060 PRINT " ***                      ***"
6070 PRINT " *** あなた の コンテ゛ィション は ? ***"
6080 PRINT " ***                      ***"
6090 PRINT " ****************************"
6100 RETURN
```

```
10000 '
10010 '*********** にゅうりょく ルーチン
10020 '
10030 '----------- うまれた 日
10040 A$=""
10050 LOCATE 2,8:PRINT SPC(26)
10060 LOCATE 2,8:INPUT "あなた の うまれは せいれき なん年 ";A$
10070 Y1=VAL(A$)
10080 IF Y1=<0 THEN 10050
10090 LOCATE 2,10:PRINT SPC(26)
10100 LOCATE 2,10:INPUT"なん月 ";A$
10110 M1=VAL(A$)
10120 IF M1<1 OR M1>12 THEN 10090
10130 LOCATE 2,12:PRINT SPC(26)
10140 LOCATE 2,12:INPUT"なん日 ";A$
10150 D1=VAL(A$)
10160 IF D1<1 OR D1>31 THEN 10130
10170 LOCATE 0,8
10180 FOR I=1 TO 12
10190  PRINT SPC(28)
10200 NEXT I
12000 '----------- しらべる 日
12010 A$=""
12020 LOCATE 2,8:PRINT SPC(26)
12030 LOCATE 2,8:INPUT "しらべたい 日 は せいれき なん年 ";A$
12040 Y2=VAL(A$)
12050 IF Y2=<0 THEN 12020
12060 LOCATE 2,10:PRINT SPC(26)
12070 LOCATE 2,10:INPUT"なん月 ";A$
12080 M2=VAL(A$)
12090 IF M2<1 OR M2>12 THEN 12060
12100 LOCATE 2,12:PRINT SPC(26)
12110 LOCATE 2,12:INPUT"なん日 ";A$
12120 D2=VAL(A$)
12130 IF D2<1 OR D2>31 THEN 12100
12140 RETURN
20000 '
20010 '*********** にっすう に へんかん
20020 '
20030 YY=Y1:MM=M1:DD=D1
20040 GOSUB 65000:'======> にっすう へんかん へ
20050 D1=DA
20060 '----------- せいぞん にっすう
20070 YY=Y2:MM=M2:DD=D2
20080 GOSUB 65000:'======> にっすう へんかん へ
20090 DI=DA-D1
```

```
20100 '---------- ハ゛イオち の けいさん
20110 DA=DI-DD
20120 RETURN
30000 '
30010 '********** しゅうき あまりけいさん ルーチン
30020 '
30030 FOR I=1 TO 3
30040  DA(I)=DA-CY(I)*INT(DA/CY(I))
30050 NEXT I
30060 RETURN
40000 '
40010 '********** ひょうし゛ と けいさん
40020 '
40030 SCREEN 2:COLOR 15,1,1:CLS
40040 OPEN "GRP:" FOR OUTPUT AS#1
40050 PRESET(100,0):PRINT#1,"ハ゛イオリス゛ム"
40060 PRESET(10,20):PRINT#1,Y2;"年";M2;"月";D2;"日は "
40070 PRESET(10,30):PRINT#1," うまれてから";DI;"日めて゛す。"
40080 LINE(180,20)-(200,20),13
40090 PRESET(220,16):PRINT#1,"しんたい"
40100 LINE(180,30)-(200,30),2
40110 PRESET(205,26):PRINT#1,"かんし゛ょう"
40120 LINE(180,40)-(200,40),7
40130 PRESET(230,36):PRINT#1,"ちせい"
40140 '
40150 '---------- わくく゛み を かく
40160 '
40170 PRESET (15,55):PRINT#1,"1"
40180 FOR X=1 TO 31
40190  LINE(X*7+8,68)-(X*7+8,190),15
40200  IF X MOD 5=0 THEN PRESET(X*7-4,55):PRINT#1,X
40210 NEXT X
40220 LINE(15,130)-(225,130),15
40230 LINE(15,69)-(225,190),15,B
40240 '
40250 '---------- かーふ゛ を かく
40260 '
40270 FOR I=1 TO 3
40280  PSET(15,130-60*SIN(2*3.14159*(DA(I))/CY(I)))
40290  FOR J=1 TO 31 STEP .5
40300   Y=130-60*SIN(2*3.14159*(J+DA(I))/CY(I))
40310   LINE -(J*7+8,Y),COL(I)
40320  NEXT J
40330 NEXT I
40340 RETURN
```

```
50000 ′
50010 ′************ けいぞ゛く はんてい まち
50020 ′
50030 CO=0
50040 PRESET(10,40)
50050 PRINT#1,"ほか の 日 を しらべ゛ますか ? "
50060 A$=INPUT$(1)
50070 IF A$="y" THEN CO=1
50080 CLOSE#1
50090 SCREEN 1:WIDTH 29
50100 RETURN
60000 ′
60010 ′********** て゛ーた
60020 ′
60030 DATA 0,31,59,90,120,151,181,212,243,273,304,334
60040 DATA 23,28,33
60050 DATA 13,2,7
65000 ′
65010 ′********** にっすう けいさん うるう年 ほせい
65020 ′
65030 DA=YY*365+M(MM)+DD
65040 DA=DA+INT(((YY-1)/4)+1)
65050 DA=DA-INT(((YY-1)/100)+1)
65060 DA=DA+INT(((YY-1)/400)+1)
65070 IF (MM>2) AND (YY/4=INT(YY/4)) THEN DA=DA+1
65080 RETURN
```

APPENDIX

1 キャラクタコード表

(1) キャラクタコードの表

3-7および3-8で説明しているとおり，MSXで使用する文字や記号には，それぞれ番号(キャラクタコード)が付いています．このキャラクタコードの表を載せておきますので，プログラムを作るときの参考にしてください．

番号	文字	番号	文字	番号	文字	番号	文字
0	↓	32	スペース	64	@	96	`
1		33	!	65	A	97	a
2		34	″	66	B	98	b
3		35	#	67	C	99	c
4		36	$	68	D	100	d
5		37	%	69	E	101	e
6		38	&	70	F	102	f
7		39	′	71	G	103	g
8		40	(	72	H	104	h
9		41	)	73	I	105	i
10		42	*	74	J	106	j
11		43	+	75	K	107	k
12		44	,	76	L	108	l
13		45	−	77	M	109	m
14	制御用コード	46	.	78	N	110	n
15		47	/	79	O	111	o
16		48	0	80	P	112	p
17		49	1	81	Q	113	q
18		50	2	82	R	114	r
19		51	3	83	S	115	s
20		52	4	84	T	116	t
21		53	5	85	U	117	u
22		54	6	86	V	118	v
23		55	7	87	W	119	w
24		56	8	88	X	120	x
25		57	9	89	Y	121	y
26		58	:	90	Z	122	z
27		59	;	91	[	123	{
28		60	<	92	¥	124	¦
29		61	=	93	]	125	}
30		62	>	94	^	126	~
31	↓	63	?	95	_	127	

※キャラクタコードの0から31は，画面上に文字の表れない特殊なキーや機能に対応している．本書で扱った制御用コードは右の通り．

コード	キー
13	RETURN
28	⇨
29	⇦
30	⇧
31	⇩

番号	文字	番号	文字	番号	文字	番号	文字
128	♠	160		192	タ	224	た
129	♥	161	。	193	チ	225	ち
130	♣	162	「	194	ツ	226	つ
131	◆	163	」	195	テ	227	て
132	○	164	、	196	ト	228	と
133	●	165	・	197	ナ	229	な
134	を	166	ヲ	198	ニ	230	に
135	ぁ	167	ァ	199	ヌ	231	ぬ
136	ぃ	168	ィ	200	ネ	232	ね
137	ぅ	169	ゥ	201	ノ	233	の
138	ぇ	170	ェ	202	ハ	234	は
139	ぉ	171	ォ	203	ヒ	235	ひ
140	ゃ	172	ャ	204	フ	236	ふ
141	ゅ	173	ュ	205	ヘ	237	へ
142	ょ	174	ョ	206	ホ	238	ほ
143	っ	175	ッ	207	マ	239	ま
144		176	ー	208	ミ	240	み
145	あ	177	ア	209	ム	241	む
146	い	178	イ	210	メ	242	め
147	う	179	ウ	211	モ	243	も
148	え	180	エ	212	ヤ	244	や
149	お	181	オ	213	ユ	245	ゆ
150	か	182	カ	214	ヨ	246	よ
151	き	183	キ	215	ラ	247	ら
152	く	184	ク	216	リ	248	り
153	け	185	ケ	217	ル	249	る
154	こ	186	コ	218	レ	250	れ
155	さ	187	サ	219	ロ	251	ろ
156	し	188	シ	220	ワ	252	わ
157	す	189	ス	221	ン	253	ん
158	せ	190	セ	222	゛	254	
159	そ	191	ソ	223	゜	255	

(2) 特別なキャラクタコードの表

グラフィックキャラクタの一部に，(1)の表に載っていないものがあります．これらは２文字分のコードを必要とする特別な文字で，CHR$(1)と組み合わせて表します．これらのキャラクタコード表を右に載せておきます．

※この表に挙げる文字や記号を表すには，２文字分のコードが必要となる．そのひな形は次のとおり．
　表中の番号ｎの文字を表す
　→ CHR$(1)＋CHR$(n＋64)
例)「秒」を画面に表示する
　PRINT　CHR$(1)＋CHR$(76)

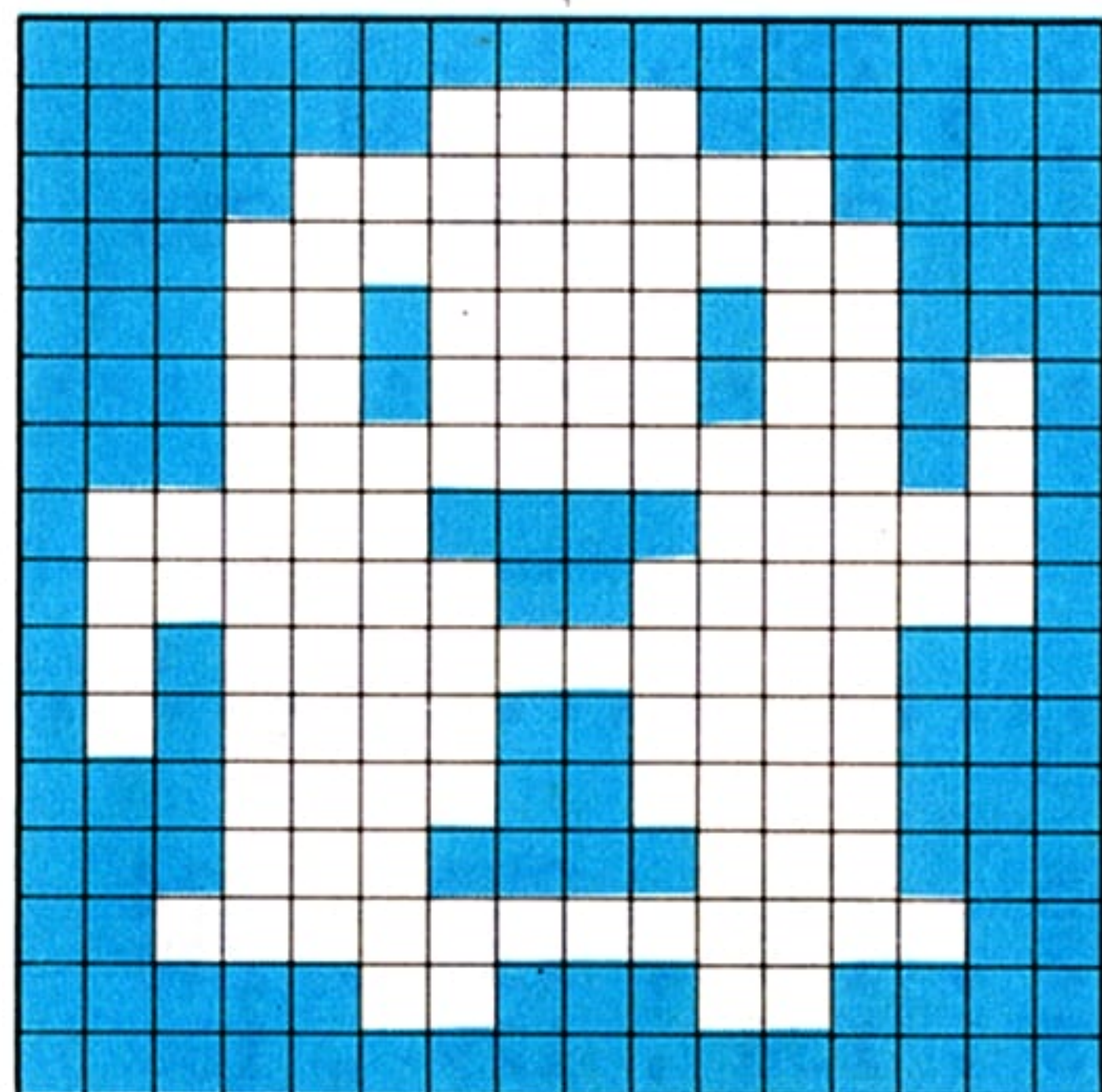
番号	文字	番号	文字
0		16	π
1	月	17	┴
2	火	18	┬
3	水	19	┤
4	木	20	├
5	金	21	┼
6	土	22	│
7	日	23	─
8	年	24	┌
9	円	25	┐
10	時	26	└
11	分	27	┘
12	秒	28	×
13	百	29	大
14	千	30	中
15	万	31	小

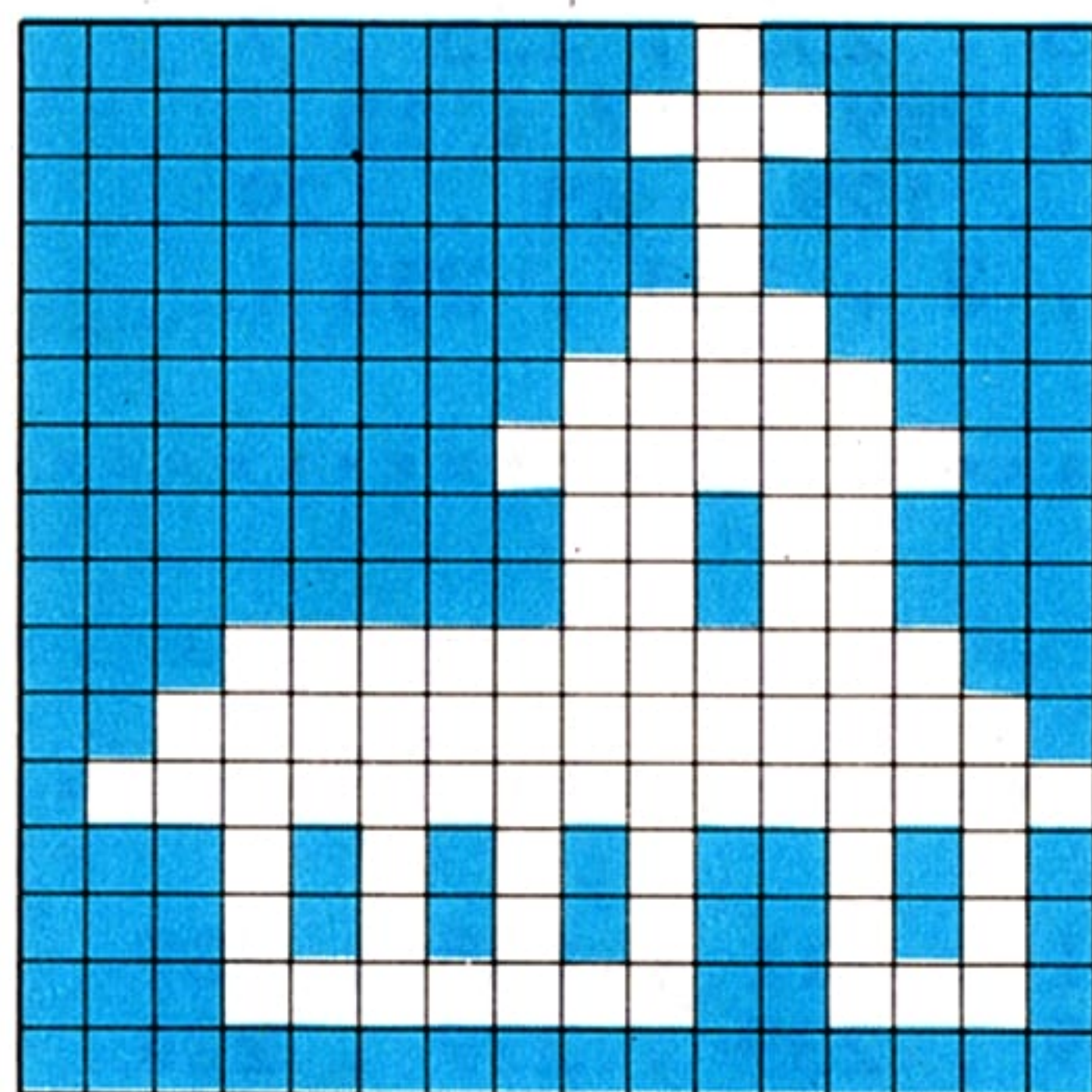
2 計算に役立つ主な記号と関数

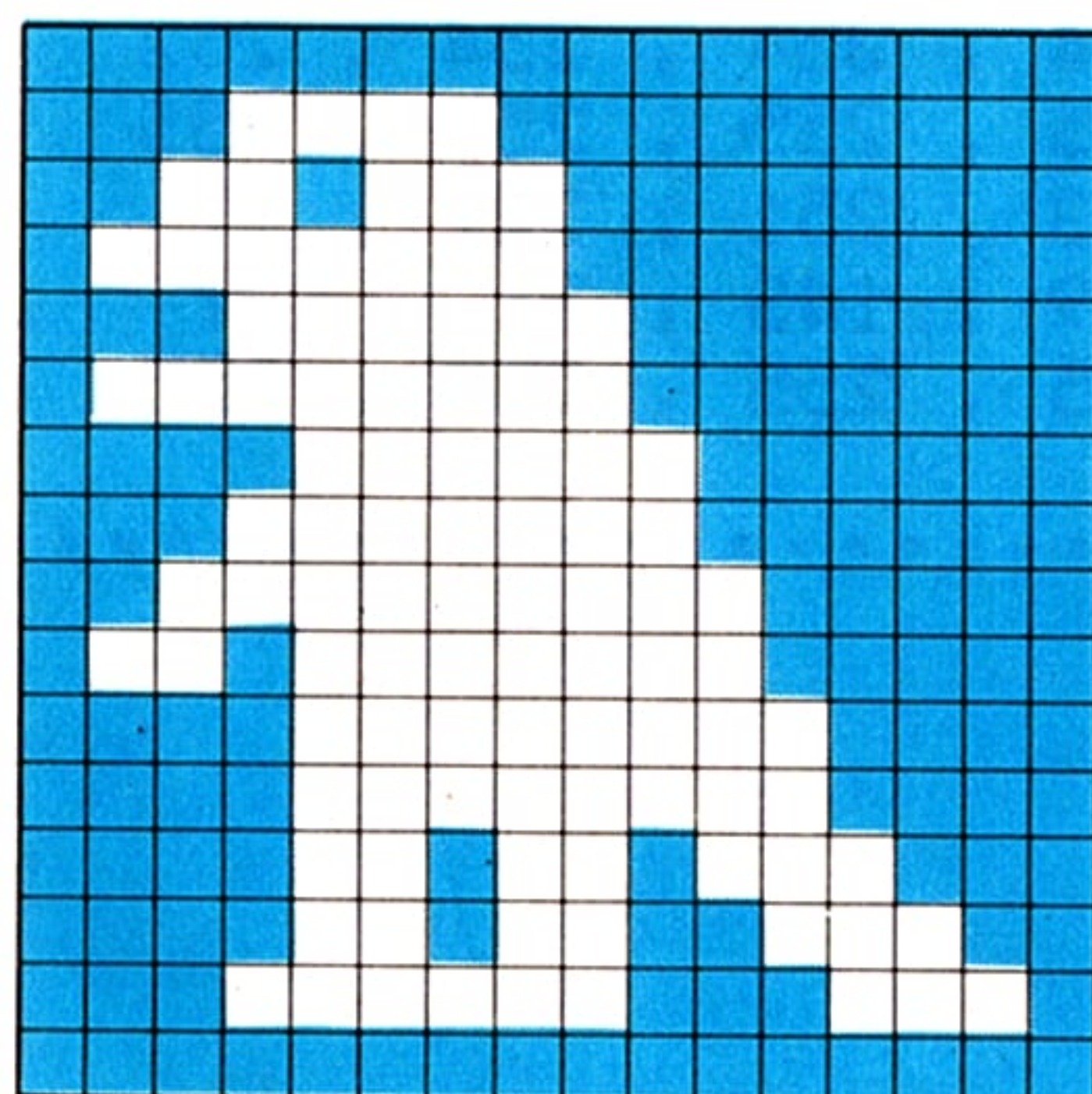
数学で利用する演算用の記号や関数は，MSXにも用意されています．主なものを右の表に挙げますので，利用してください．

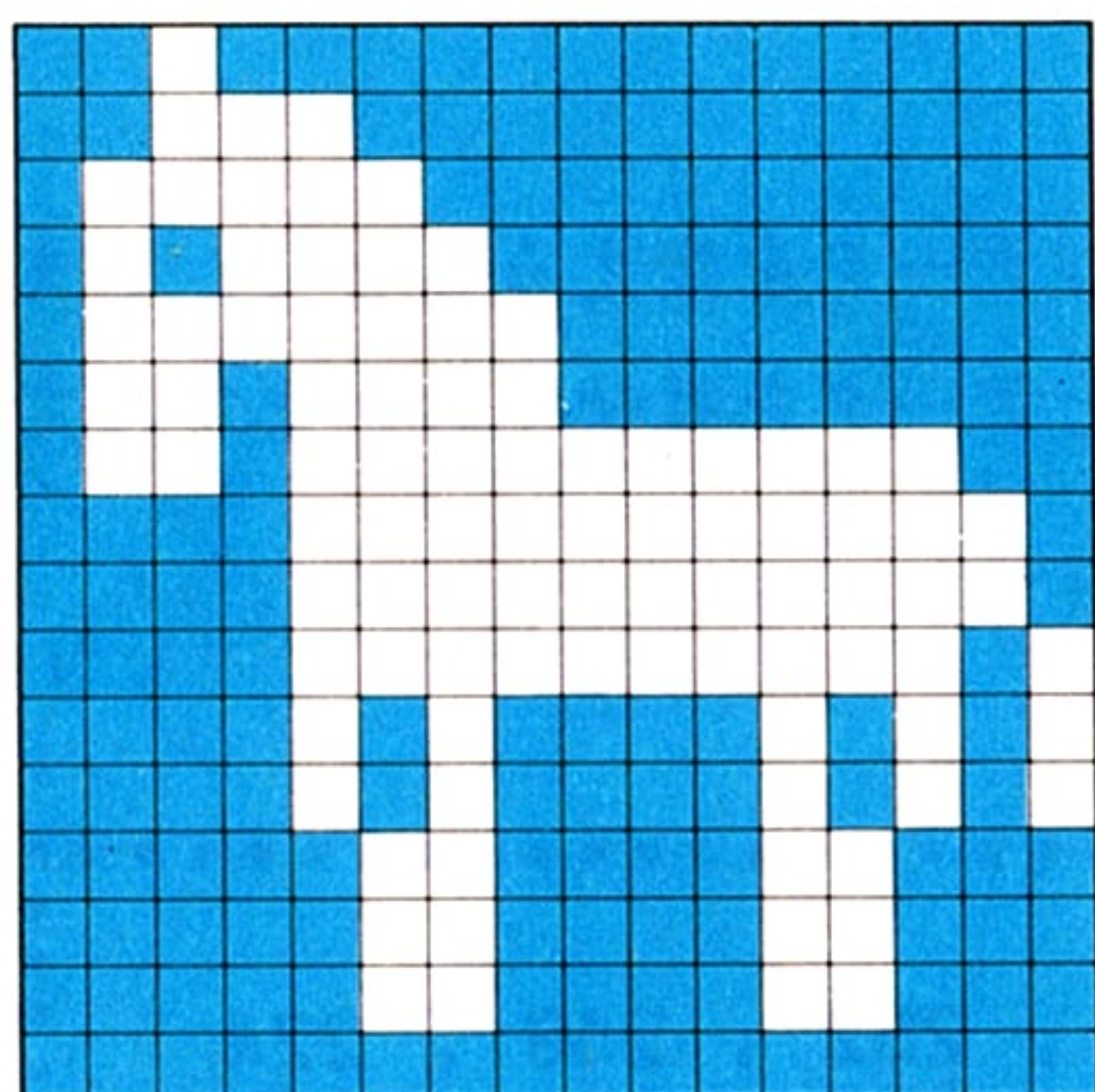
MSXでの記号・関数	意味	例
＝	代入	A＝B
＋	＋	A＝B＋C
−	−	A＝B−C
＊	×	A＝B＊C
／	÷	A＝B／C
¥	整数の割り算	A＝B¥C
MOD	あまり	A＝B MOD C
∧	べき乗	A＝B∧C
SIN	サイン	A＝SIN(B)
COS	コサイン	A＝COS(B)
TAN	タンジェント	A＝TAN(B)
FIX	整数部をとりだす	A＝FIX(B)
INT	小数を切り捨てた整数値	A＝INT(B)
LOG	自然対数	A＝LOG(B)
EXP	指数関数	A＝EXP(B)

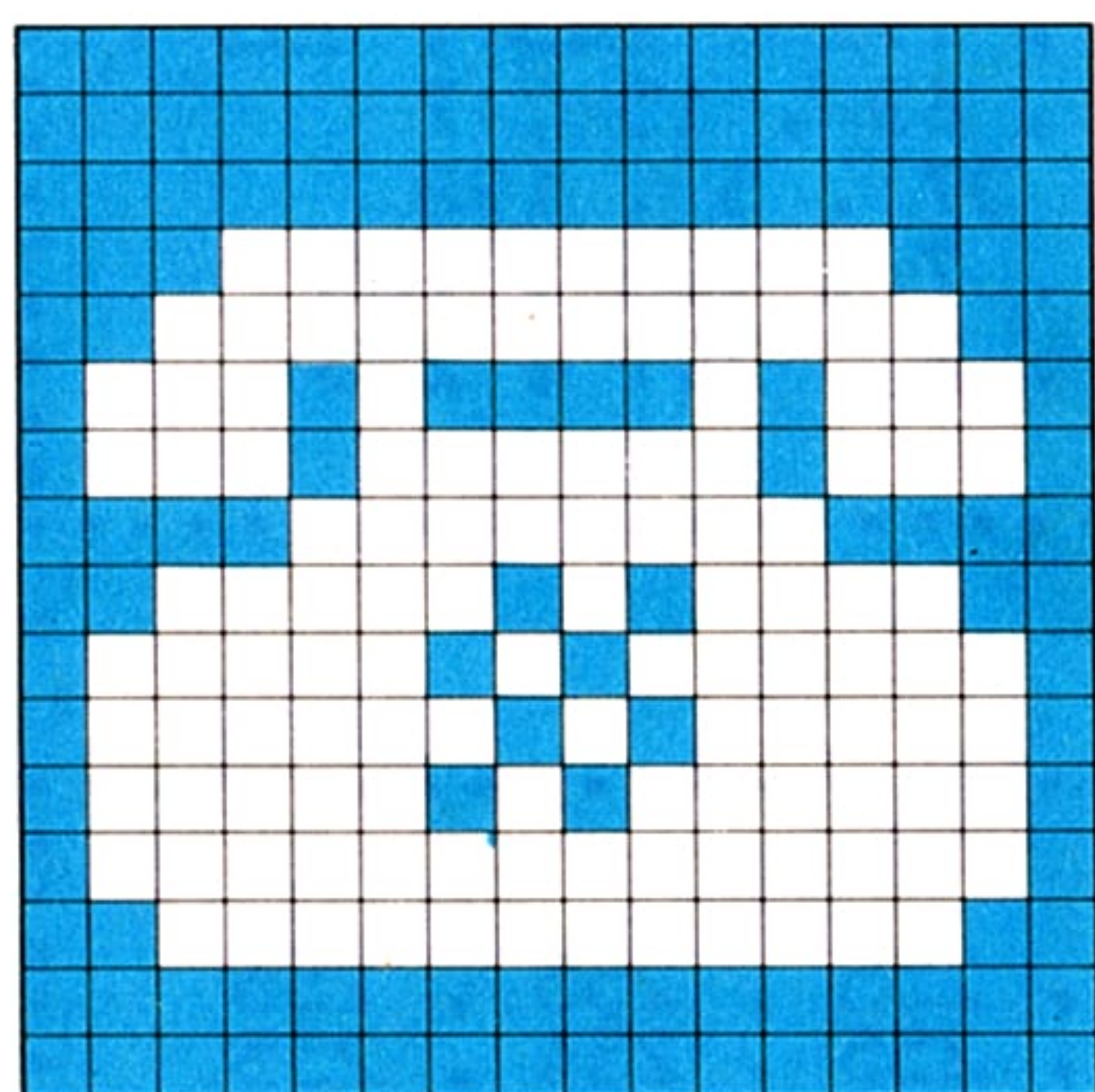
③ スプライトパターンのサンプルデータ

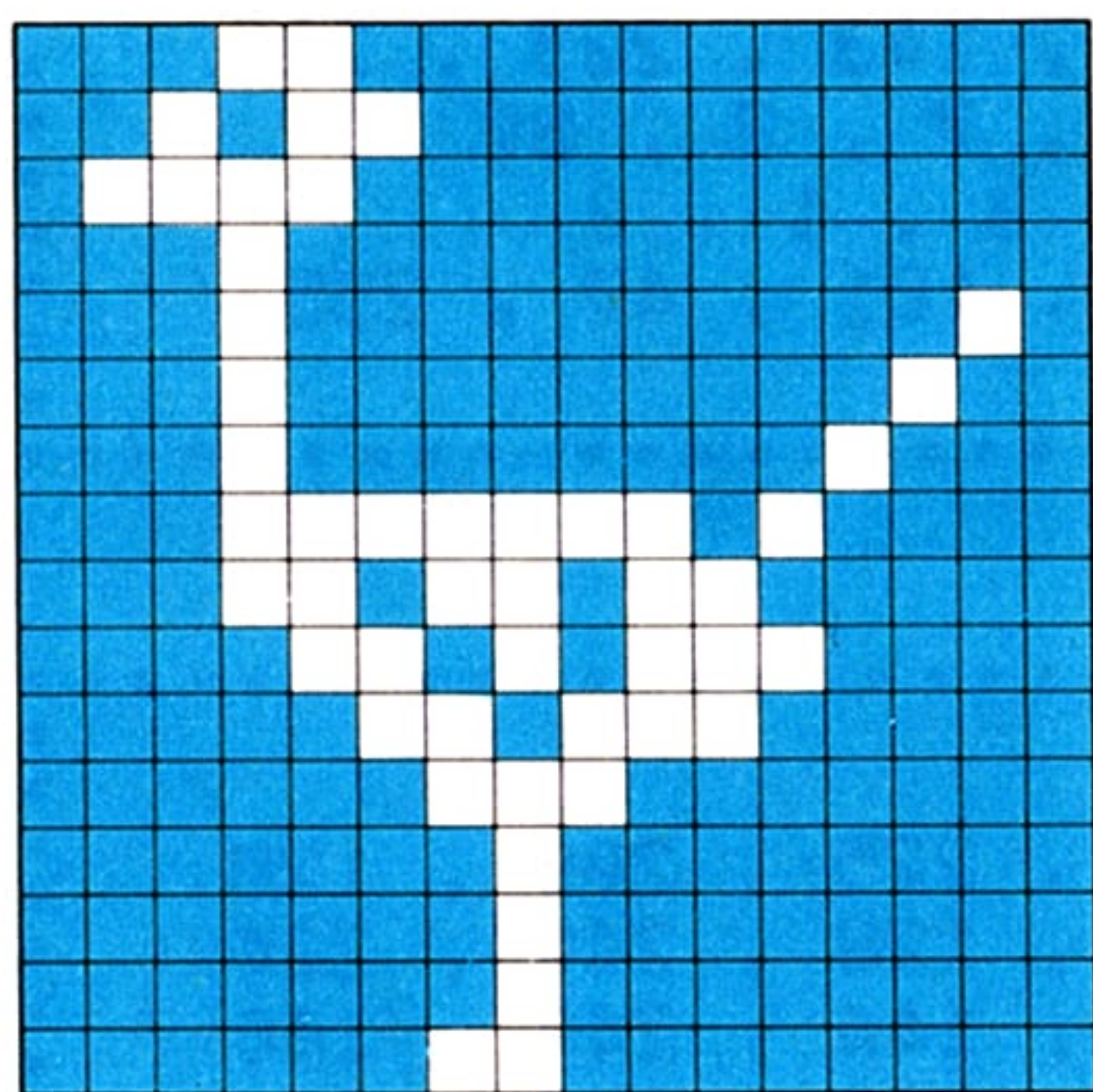
4-4のスプライトを使ったスロットマシンのデザインは,データを変えることで, さまざまな形に設定することができます. サンプルデザインとデータを載せておきますので, 1010行以後のデータを変更して, 遊んでください.

■ ペンギン ■

```
data  255 , 252 , 240 , 224
data  228 , 228 , 224 , 131
data  129 , 160 , 161 , 225
data  227 , 192 , 243 , 255
data  255 , 63 , 15 , 7
data  39 , 37 , 5 , 193
data  129 , 7 , 135 , 135
data  199 , 3 , 207 , 255
```

■ キョウカイ ■

```
data  255 , 255 , 255 , 255
data  255 , 255 , 254 , 255
data  255 , 224 , 192 , 128
data  234 , 234 , 224 , 255
data  223 , 143 , 223 , 223
data  143 , 7 , 3 , 39
data  39 , 3 , 1 , 0
data  181 , 181 , 49 , 255
```

■ ゴジラ ■

```
data  255 , 225 , 200 , 128
data  224 , 128 , 240 , 224
data  192 , 144 , 240 , 240
data  242 , 242 , 224 , 255
data  255 , 255 , 255 , 255
data  127 , 127 , 63 , 63
data  31 , 31 , 15 , 15
data  71 , 99 , 113 , 255
```

■ウマ■

```
data  223 , 199 , 131 , 161
data  128 , 144 , 144 , 240
data  240 , 240 , 245 , 245
data  249 , 249 , 249 , 255
data  255 , 255 , 255 , 255
data  255 , 255 , 3 , 1
data  1 , 2 , 234 , 234
data  231 , 231 , 231 , 255
```

■デンワ■

```
data  255 , 255 , 255 , 224
data  192 , 139 , 136 , 240
data  193 , 130 , 129 , 130
data  128 , 192 , 255 , 255
data  255 , 255 , 255 , 7
data  3 , 209 , 17 , 15
data  67 , 129 , 65 , 129
data  1 , 3 , 255 , 255
```

■ダチョウ■

```
data  231 , 211 , 135 , 239
data  239 , 239 , 239 , 224
data  228 , 242 , 249 , 252
data  254 , 254 , 254 , 252
data  255 , 255 , 255 , 255
data  253 , 251 , 247 , 47
data  159 , 143 , 31 , 127
data  255 , 255 , 255 , 255
```

4 MSX BASIC索引

命令	プログラムの作成・実行	プログラムの流れの制御	出力				入力		その他	機能	CHAPTER - section	ページ
			文字	グラフィックス	音	カセット	カセット	キーボード				
ASC									●	文字コードを求める	3－8	116
AUTO	●									行番号の自動発生	2－4	52
CHR$									●	文字コード → 文字	3－7	109
CIRCLE				●						円を描く	2－1	34
CLOAD							●			プログラムを読み込む	2－6	67
CLOAD?							●			プログラムの確認	2－6	65
CLOSE						●	●			OPENした入出力装置の使用終了	3－9	121
CLS			●	●						画面を消す	3－3	84
COLOR			●	●						色を変える	2－3	45
CONT	●									プログラムの再開	3－4	93
CSAVE						●				プログラムの書き込み	2－6	64
DATA									●	データを用意する	4－2	133
DELETE	●									行の削除	3－2	78
DIM									●	配列の使用宣言	3－5	96
DRAW				●						線を描く	4－5	150
END		●								プログラムの終了	3－1	75
FOR~NEXT~STEP		●								繰り返しの設定	2－4	51
GOSUB／RETURN		●								サブルーチン	4－4	146
GOTO		●								任意の行番号を実行	2－5	60
IF～THEN～ELSE		●								条件判定	3-1／3-8	73/115
INKEY$								●		キーボードからリアルタイムで読み込む	3－3	85
INPUT								●		キーボードから入力	2－2	41
INPUT ♯							●			テープから読み込む	3－9	121
INPUT$								●		キーボードから任意の文字数を読み込む	4－5	153
INT									●	小数部分の切り捨て	2－5	58

命　令	プログラムの作成・実行	プログラムの流れの制御	出　力				入力		その他	機　能	CHAPTER | section	ページ
			文字	グラフィックス	音	カセット	カセット	キーボード				
KEY LIST									●	ファンクションキーの内容表示	2 − 3	49
LEFT$									●	文字列の一部を取り出す(左)	3 − 7	110
LEN									●	文字列の長さを調べる	3 − 7	107
LINE				●						線，四角を描く	4 − 6	159
LINE INPUT								●		キーボードから入力	3 − 8	117
LIST	●									リストを表示する	2 − 1	33
LOCATE			●							文字表示位置の指定	3 − 3	83
MID$									●	文字列の一部を取り出す(中)	3 − 8	115
MOTOR									●	テープのモータのON/OFF	2 − 6	65
NEW	●									プログラムの消去	2 − 1	32
ON～GOTO		●								条件により指定した行番号から実行	4 − 3	140
OPEN				●		●	●			カセット・グラフィック画面の使用宣言	3-9／4-6	121/157
PAINT				●						色を塗る	2 − 3	47
PLAY					●					音を出す	4 − 7	162
PRESET				●						点を消す	5 − 4	195
PRINT(?)			●							文字・数字の表示	3 − 1	72
PRINT♯				●		●				カセット・グラフィック画面への書き込み	3-9／4-6	121/157
PSET				●						点を打つ	4 − 6	157
PUT SPRITE				●						スプライトの表示	4 − 3	136
READ									●	プログラム内のデータの読み込み	4 − 2	134
REM(')									●	コメント	3 − 1	71
RENUM	●									行番号の付け直し	2-2／4-4	38/142
RIGHT$									●	文字列の一部を取り出す(右)	3 − 7	110
RND									●	乱数の発生	2 − 5	57
RUN	●									プログラムの実行	2 − 1	33

命令	プログラムの作成・実行	プログラムの流れの制御	出力				入力		その他	機能	CHAPTER－section	ページ
			文字	グラフィックス	音	カセット	カセット	キーボード				
SCREEN			○	○						画面モード・スプライトの設定	4-1／4-2	126/132
SOUND					○					音を出す	4－9	178
SPC			○							空白を表示する	5－3	192
SPRITE$				○						スプライトの定義	4－2	135
STICK								○		ジョイスティック・カーソルの状態を示す	4－3	139
STRING$									○	同一文字を並べた文字列を作る	3－6	104
TIME									○	タイマー	2-5／3-1	61/71
TRON	○									トレースオン	3－5	99
TROFF	○									トレースオフ	3－5	99
VAL									○	文字列の数値化	4－2	135

MSXビギナーズBASIC

1984年3月20日　初版発行
定価1,500円

著　者　児玉真之（こだまさねゆき）
発行者　塚本慶一郎
発行所　株式会社アスキー
　〒107　港区南青山5－11－5住友南青山ビル
振　替　東京4-161144
電　話　03―486―7111（代表）

編集担当　土田米一
表紙担当　郷　啓子
印刷　壮光舎印刷株式会社

ISBN4-87148-729-6 C3055 ¥1500E